MASTER EN SCIENCES INFORMATIQUES

Applications de la programmation déclarative au problème de planification des visites médicales des Services de Promotion de la Santé à l'Ecole

Rwanyindo, Clément

Award date:
2019

Awarding institution:
Universite de Namur

Université de Namur
Faculté d'informatique
Année académique 2017-2018

Application de la programmation déclarative au problème de planification des visites médicales des Services de Promotion de la Sante à l'Ecole

Clément Rwanyindo

Promoteur : Jean-Marie JACQUET.
(Signature pour approbation du dépôt - REE art. 40)

Mémoire présenté en vue de l'obtention du grade de
Master en Sciences Informatique

Résumé

Nous analysons dans ce mémoire les possibilités de développement d'une application de planification des rendez-vous des visites médicales effectuées par les Services de Promotion de la Santé à l'Ecole auprès des élèves des établissements liés à ces services. Nous développons ensuite une solution au travers de la programmation déclarative.

Abstract

In this memoir, we analyze the possibilities of creating an application for the foster healthcare services at school. This application would facilitate the planning of medical examinations with the students of all schools linked to each services. We then develop a solution using declarative programming.

Remerciements

Ce mémoire n'aurait pu être réalisé sans le concours de nombreuses personnes.

Ma femme, Candy PETIT, qui a prise sur elle pour que je dispose d'un maximum de temps possible pour la rédaction, les recherches et les rendez-vous menées dans le cadre de ce travail alors que cette année fut plus que chargée.

Mes enfants, Noé (3 ans) et Emmie (20 mois), encore incapable de réaliser à quelle point cela m'a peiné de leur infliger mes absences alors qu'ils ont été de tels amours en intégrant et acceptant l'idée que « papa devait aller travailler ».

Mes parents qui, en plus de la merveilleuse guidance qu'ils m'ont offert tout au long de ma vie, ont à maintes reprises trouvé le temps et l'énergie d'apporter à mon cocon familiale le soutien psychologique et logistique durant tout ce cursus. Ainsi que mes beaux-parents pour toute l'aide fournie durant la réalisation de ce mémoire.

Les membres du service de Promotion de la Santé à l'Ecole de Namur, Mesdames Brigitte COLLET, Nathalie COLOT, Charlotte MOURIN et Marianne BODSON pour les entrevues accordées et les réponses de suivi au mail de demandes d'informations.

Les divers PMS et PSE qui ont fait suite à mes courriels de demandes d'informations.

Mon ami de longue date, Patrick KALUME, étudiant de Master en Sciences Informatique également, pour nos nombreux échanges concernant les cours et nos travaux respectifs à réaliser.

Enfin, mon promoteur, Monsieur Jean-Marie JACQUET pour avoir accepté de suivre la réalisation de ce mémoire, pour ses nombreux conseils et pour le temps libéré pour faire suite à mes questions.

Table des matières

Préface

Dans ce mémoire nous nous intéressons à l'application de la programmation déclarative pour résoudre un problème de planification.

La programmation déclarative est un paradigme de programmation au même titre que la programmation impérative ou encore la programmation orientée objet pour ne citer que les plus usités et connus du grand public[1].

C'est lors de mon cursus de master en sciences informatiques que la programmation déclarative m'a été introduite.

Au-delà de l'énorme surprise qu'un tel domaine de programmation puisse m'être encore inconnu, ce fut un réel challenge que de devoir repenser et adopter une nouvelle approche à la résolution de problèmes informatiques.

Plus frappant encore, ce paradigme permet de résoudre certaines problématiques à la fois plus rapidement et avec souvent un code plus concis.

Durant l'un des cours du cursus, un des exemples d'application de la programmation déclarative énoncé par le professeur concernait l'élaboration d'un horaire dans un établissement scolaire.

Un horaire devant prendre en compte les classes d'élèves existantes, les attributions pédagogiques des professeurs et les ressources disponibles (locaux), il semble de prime abord assez complexe d'aboutir à un résultat exempt de conflits.

Quelle ne fut pas ma surprise d'entendre le professeur nous assurer que la programmation déclarative, de par sa nature, permet de résoudre le problème de manière plus efficiente.

Intrigué depuis, l'idée de réaliser mon mémoire de fin d'étude sur un cas semblable a, petit à petit, germé dans mon esprit.

Dans le cadre de mon activité professionnel, j'ai été amené à collaborer avec des Services de Promotion de la Santé à l'Ecole (P.S.E) dans le but de fournir à ces derniers un nouveau logiciel répondant à leur besoin métier, à savoir l'encodage des bilans de santé des élèves rencontrés lors des visites médicales.

Ces services PSE sont amenés à planifier des rendez-vous avec des classes de divers établissements afin de rencontrer les élèves pour effectuer des visites médicales.

Face à l'énorme tâche que constitue la planification de tous ces rendez-vous, la question que l'on peut se poser est « Est-il possible, dans ce cadre, de mettre en place un outil de planification automatisé à l'aide d'une technique informatique et plus particulièrement de la programmation déclarative ? »

Ce mémoire tâche de répondre à cette question. Il est composé de quatre parties.

La première partie vise à définir la problématique à traiter en étudiant son contexte et les solutions utilisées actuellement. La seconde partie dresse un état de l'art des techniques de planification existantes. La troisième partie fournit une solution au problème de planification des rendez-vous des PSE. Enfin, nous dressons les conclusions tirés de notre étude dans la quatrième partie.

[1] Entendre ici public d'informaticien

Ce mémoire s'inscrit dans la catégorie des mémoires appliqués et vise à résoudre une problématique de planification à l'aide d'une méthode de programmation dite déclarative.

Nous privilégions tout au long de ce travail l'adoption d'une approche et démarche scientifique au travers de méthodologies empiriques. De fait, les interviews et études de la réalité de terrain prendront une place importante et viendront compléter la recherche documentaire aussi bien pour la partie métier que l'aspect technique avant application de la technique de résolution envisagée.

Cette approche vise à développer un esprit critique et objectif ainsi qu'une méthodologie de travail qui pourra être appliqué à la résolution d'autres problèmes informatiques.

Chapitre 1 La planification des visites médicales organisées par les Services de Promotion de la Santé à l'Ecole

Initialement appelé « Inspection Médicale Scolaire (IMS) »[1], c'est en 1964 que le programme de dépistage et de lutte contre les maladies infectieuses fut créé. Rendu obligatoire tout en étant gratuit dans tous les établissements, il permettait également de détecter les déficiences physiques ou mentales des élèves [8].

L'objectif de ce chapitre est de définir le cadre juridique et administratif régissant les PSE, de rappeler ses missions et, au travers notamment d'une enquête menée sur le terrain, de cerner la problématique de l'organisation des visites médicales.

1.1. Les décrets

1.1.1. Le décret de 1997

La Communauté Française adopte en 1997 un nouveau décret mettant à jour l'encadrement et le suivi médical en milieu scolaire. Le principal changement concerne l'adoption d'une vision plus générale de cet encadrement et suivi des élèves en allant plus loin que de simplement considérer les aspects médicaux ;

« Par santé au sens du présent décret, il faut entendre un bien-être physique, mental et social ; la santé ne consistant pas seulement en une absence de maladie ou d'infirmité. »[2]

« Par promotion de la santé au sens du présent décret, il faut entendre le processus qui vise à permettre à l'individu et à la collectivité d'agir sur les facteurs déterminants de la santé et, ce faisant, d'améliorer celle-ci… »[2].

Les rôles et missions des différents intervenants (conseil supérieur de la santé, …) sont également détaillés ainsi que la définition de l'organisation du programme de promotion de la santé au travers d'un plan d'action quinquennal ainsi que les objectifs visés et l'évaluation de ces derniers.

1.1.2. Les décrets de 2001 et 2002

En 2001 et 2002, de nouveaux décrets relatifs à la promotion de la santé respectivement à l'école et dans l'enseignement supérieur hors université sont adoptés.

Le Conseil de la Communauté française détaille dans ces décrets plus précisément les missions (cf. section suivante) ayant trait à la promotion de la santé à l'école.

La finalité de cette mission de santé public, outre l'accompagnement médical de la partie de population que sont les élèves, consiste également à recueillir et à analyser des données statistiques afin d'aiguiller les décisions politiques et de confirmer ou réorienter les choix stratégiques antérieures.

L'appellation « Promotion de la Santé à l'Ecole » est adoptée en remplacement de « Inspection Médicale Scolaire ».[3]

1.2. Les missions

« La promotion de la santé à l'école à l'école consiste en :

1. La mise en place de programmes de promotion de la santé et de promotion d'un environnement scolaire favorable à la santé ;
2. Le suivi médical des élèves, qui comprend les bilans de santé individuels et la politique de vaccination, tel que précisé à l'article 6 ;
3. La prophylaxie et le dépistage des maladies transmissibles, telles que précisées à l'article 7 ;

4. L'établissement d'un recueil standardisé de données sanitaires, tel que précisé à l'article 8.

»[4]

De ces 4 lignes stratégiques encadrant le travail des services de promotion de la santé, le second, « **le suivi médical des élèves** », **constitue 70% de la charge de travail** des membres du service, point également définit dans le décret de 2001.

1.3. Les partenaires

Les services de promotions de la santé font partie d'un écosystème administratif et sont amenés à interagir avec de nombreuses parties. On retrouve parmi ces parties les acteurs suivants :

- Le pouvoir organisateur
- La direction des services de promotions de la santé
- Les centres psycho-médicaux-sociaux (PMS)
- Les établissements scolaires
- Les parents des élèves ou les élèves eux-mêmes s'ils sont majeurs

D'autres acteurs existes tes que la fédération Wallonie-Bruxelles, etc. mais nous n'allons pas en parler ici car sans impact direct pour la problématique visée.

1.3.1. Le pouvoir organisateur

Le **pouvoir organisateur** d'un service de promotion de la santé (ou d'un établissement scolaire) est « l'autorité, la ou les personne(s) physique(s) ou morale(s) ou privée(s), qui en assume(nt) la responsabilité »[5].

Le pouvoir organisateur d'une entité (PSE, établissement, PMS, …) a en fait à sa charge la gestion administrative de cette entité. Il définit les modes de fonctionnement, gère les relations avec des entités partenaires (convention entre un PSE et une école, …) et opère des choix stratégiques (informatisation, restructuration, …).

Les décisions managériales du pouvoir organisateur ne peuvent cependant pas aller à l'encontre des missions des entités constituées. De fait, leur missions sont régies par des lois et restent le cœur même du métier de ces entités.

1.3.2. La direction des services de promotion de la santé

Si les Services de Promotion de la Santé à l'Ecole (PSE) voient leur gestion interne (Ressources Humaines, Informatisation, …) définie par leur pouvoir organisateur, c'est la direction des services de promotions de la santé, organe (cf. annexe 2) de l'Office de la Naissance et de l'Enfance (ONE) qui encadre leur mission.

Initialement la promotion de la santé était gérée par la Direction Générale de la Santé en Fédération Wallonie-Bruxelles. C'est suite à la 6ème réforme de l'état que l'ONE s'est vu attribuer de nouvelles compétences en matière de santé dont « l'accompagnement et le subventionnement des services de Promotion de la Santé à l'Ecole (PSE) »[6].

L'ONE, Office de la Naissance et de l'Enfance, est un organisme de la Fédération Wallonie-Bruxelles fournissant un ensemble de services de suivi des enfants tel que :

- Les consultations prénatales (visites préconceptionnelles, surveillance des grossesses et soutien à la parentalité)
- Des consultations pour enfants jusqu'à leur 6 ans
- La mise à disposition de milieu d'accueil de la petite enfance

- La mise à disposition de lieu d'accueil tel que les écoles de devoirs, les centres de vacances, l'accueil extrascolaire, …
- **La promotion de la santé**, objet ce mémoire
- Des recherches scientifiques
- L'accompagnement à l'adoption
- La prévention et prise en charge de la maltraitance
- …

Tous ces services s'articulent autour de principes issus de la convention internationale des droits de l'enfant, le soutien à la parentalité, l'inclusion et la réduction des inégalités sociales.

1.3.3.Les centres psycho-médicaux-sociaux

Collaborateur privilégié des services de promotion de la santé, les centres psycho-médicaux-sociaux accompagnent les élèves et leurs parents de l'entrée en maternelle jusqu'à la fin des études secondaires avec pour missions de : «

1. Promouvoir les conditions psychologiques, psycho-pédagogiques, médicales et sociales qui offrent à l'élève les meilleures chances de développer harmonieusement sa personnalité et de le préparer à assumer son rôle de citoyens autonome et responsable et à prendre une place active dans la vie sociale, culturelle et économique ;
2. Contribuer au processus éducatif de l'élève, tout au long de son parcours scolaire, en favorisant la mise en œuvre des moyens qui permettront de l'amener à progresser toujours plus et ce, dans la perspective d'assurer à tous des chances égales d'accès à l'émancipation sociale, citoyenne et personnelle.
   A cette fin les centres mobiliseront, entre autres les ressources disponibles de l'environnement familial, social et scolaire de l'élève ;
3. Dans une optique d'orientation de son projet de vie personnelle, scolaire, professionnel et de son insertion socio-professionnelle. »[7]

Un point important à noter et que pour les établissements ayant pour pouvoir organisateur la Communauté française, c'est un PMS de ce même pouvoir qui en plus d'assurer les missions ci-dessus assurera aussi les missions de promotion de la santé.[7]

Ces PMS sont donc amenés à organiser des visites médicales et l'objet de ce mémoire pourrait leur être bénéfique également.

1.3.4.Les établissements scolaires

Le droit à l'instruction est un droit fondamental inscrit dans la constitution ainsi que dans la convention des droits de l'enfant.

En Belgique, cette instruction, obligatoire pour tous les mineurs d'âge quel que soit leur statut et ce dès lors qu'ils résident sur le territoire belge, est organisée par des établissements scolaires.

L'instruction à destination des enfants est encadrée par divers intervenants allant du directeur d'établissement, assisté dans ses missions par du personnel administratif, au corps éducatif assurant l'accompagnement et encadrement scolaire quotidien des élèves.

Le directeur de l'établissement est le responsable de l'atteinte des objectifs définis dans le projet pédagogique approuvé par son pouvoir organisateur. Il est ainsi amené à gérer l'organisation générale de son établissement, l'équipe éducative et ses attributions, les dossiers élèves, les **horaires**, l'organisation de **la collaboration avec** les services externes tels que les **PMS** et les **PSE**, etc.

Le corps éducatif est constitué d'une équipe pédagogique, les enseignants, les éducateurs, …, et d'une équipe médicale, les logopèdes, les puéricultrices, …

Toute classe d'élèves amenée à effectuer une visite médicale dans un service de promotion de la santé est accompagnée par un membre du corps éducative, le plus souvent un éducateur de l'école[2].

1.3.5. Les parents des élèves ou les élèves majeurs

Acteurs primordiaux dans le suivi de la scolarité d'un élève, les tuteurs légaux sont informés par le directeur d'établissement du centre psycho-médicaux-social et du service de promotion de la santé liés à l'établissement fréquenté par leur enfant.

1.4. Cadre légal de l'organisation des visites médicales

Un arrêté royal du Gouvernement de la Communauté française définit l'organisation générale des visites médicales.

« Les bilans obligatoires de santé, complets ou partiels, sont réalisés les années scolaires suivantes :

- dans l'enseignement maternel : en 1re et en 3e année;
- dans l'enseignement primaire : en 2e, 4e et 6e année ;
- dans le premier degré de l'enseignement secondaire : en 1re accueil, en 1re année complémentaire, en 2e générale et en 2e professionnelle ;
- dans le deuxième degré de l'enseignement secondaire : en 4e année ;
- dans l'enseignement professionnel secondaire complémentaire : en 1e année de la section « soins infirmiers » ;
- dans les centres de formation et d'éducation en alternance : la 1re année de fréquentation de ce type d'enseignement, et ensuite tous les deux ans ;
- dans l'enseignement spécialisé : la 1re année de fréquentation de ce type d'enseignement, et ensuite tous les deux ans. »[8]

Côté pratique, une annexe à cet arrêté précise certains points[8] tel que :

- Les types d'examens pratiqués[3] :
  - Le bilan partiel : anamnèse, bilan vaccinal, développement corporel et courbe de croissance, examen de la vue, de l'audition et ORL et informations recueillies auprès de l'enseignant
  - Le bilan complet : complément à l'anamnèse existant, bilan vaccinal, examen biométrique, examen clinique complet, évaluation de la psychomotricité et évolution du langage. Un questionnaire sur les habitudes de vie est rempli par les élèves du secondaire.
- L'organisation des visites durant la scolarité des élèves
  - En 1re maternelle, bilan partiel au 3e trimestre de l'année scolaire
    - bilan partiels en 2e maternelle pour les élèves n'ayant pas eu de bilan partiel en 1e
  - En 3e maternelle, on effectue un bilan complet
    - bilan complet en 1e primaire pour les élèves n'ayant pas eu de bilan complet en 3e maternelle
  - En 2e primaire, bilan complet
  - En 4e primaire, bilan partiel
  - En 6e primaire, bilan complet
  - En secondaire, le bilan complet pour est réalisé en 1re accueil, 1re année complémentaire, 1re année pour la section « soins infirmiers », 2e générale, 2e professionnelle, 4e générale ou technique de transition, 4e professionnelle ou technique de qualification

[2] Information renseignée par le PSE de Namur (notre interlocuteur de terrain) lors de l'interview

[3] Lors de l'enquête de terrain, le PSE de Namur nous informe qu'il pratique systématiquement des bilans complets

- Dans les centres d'éducation et de formation en alternance (CEFA)
- Dans l'enseignement spécialisé

1.5. Etude contextuelle

1.5.1. Quantification des intervenants

Les informations recueillies auprès de la Fédération Wallonie-Bruxelles[4] nous permettent de dresser le tableau suivant :

Intervenants	**Type**	**Nombre**
Antennes PSE		**151**
	PMS de la Communauté Française	41
	PSE	110
Antennes PMS		186
Etablissements		**4861**
	Maternel ordinaire	1763
	Maternel spécialisé	128
	Primaire ordinaire	1840
	Primaire spécialisé	170
	Secondaire ordinaire	506
	Secondaire spécialisé	95
	Secondaire artistique	112
	Centres de formation en alternance (CEFA)	207
	Hautes écoles	20
	Ecole supérieures d'architecture	4
	Ecoles supérieures des arts	16
Etablissements du supérieur	Promotion sociale[5]	165
	Universités	*7*

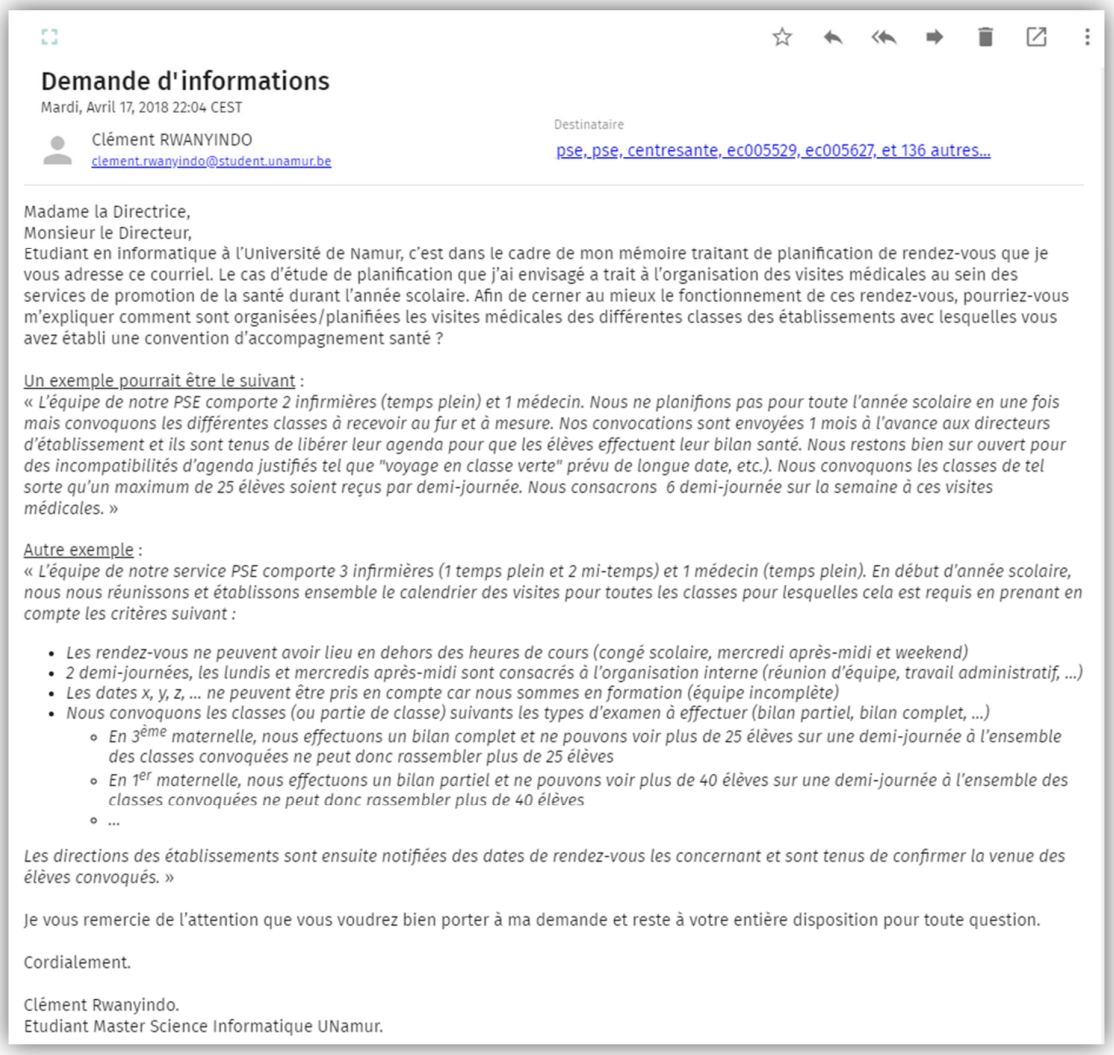
1.5.2. Enquête de terrain

Afin d'étayer les informations recueillies dans les divers décrets et arrêtés fixant l'organisation des services de promotion de la santé, des demandes de renseignements ont été introduites auprès de service pour lesquelles des informations de contact ont pu être retrouvés (141 sur les 151 antennes).

[4] http://www.enseignement.be/index.php?page=25949

[5] La promotion sociale et les universités ne font pas l'objet de suivi par les Services de Promotion de la Santé à l'Ecole

C’est suite au mail suivant que des données complémentaires ont pu être recueillies :



Demande d'informations

Mardi, Avril 17, 2018 22:04 CEST

Clément RWANYINDO
clement.rwanyindo@student.unamur.be

Destinataire
pse, pse, centresante, ec005529, ec005627, et 136 autres...

Madame la Directrice,
Monsieur le Directeur,
Etudiant en informatique à l'Université de Namur, c'est dans le cadre de mon mémoire traitant de planification de rendez-vous que je vous adresse ce courriel. Le cas d'étude de planification que j'ai envisagé a trait à l'organisation des visites médicales au sein des services de promotion de la santé durant l'année scolaire. Afin de cerner au mieux le fonctionnement de ces rendez-vous, pourriez-vous m'expliquer comment sont organisées/planifiées les visites médicales des différentes classes des établissements avec lesquelles vous avez établi une convention d'accompagnement santé ?

Un exemple pourrait être le suivant :
« L'équipe de notre PSE comporte 2 infirmières (temps plein) et 1 médecin. Nous ne planifions pas pour toute l'année scolaire en une fois mais convoquons les différentes classes à recevoir au fur et à mesure. Nos convocations sont envoyées 1 mois à l'avance aux directeurs d'établissement et ils sont tenus de libérer leur agenda pour que les élèves effectuent leur bilan santé. Nous restons bien sur ouvert pour des incompatibilités d'agenda justifiés tel que "voyage en classe verte" prévu de longue date, etc.). Nous convoquons les classes de tel sorte qu'un maximum de 25 élèves soient reçus par demi-journée. Nous consacrons 6 demi-journée sur la semaine à ces visites médicales. »

Autre exemple :
« L'équipe de notre service PSE comporte 3 infirmières (1 temps plein et 2 mi-temps) et 1 médecin (temps plein). En début d'année scolaire, nous nous réunissons et établissons ensemble le calendrier des visites pour toutes les classes pour lesquelles cela est requis en prenant en compte les critères suivant :

- *Les rendez-vous ne peuvent avoir lieu en dehors des heures de cours (congé scolaire, mercredi après-midi et weekend)*
- *2 demi-journées, les lundis et mercredis après-midi sont consacrés à l'organisation interne (réunion d'équipe, travail administratif, ...)*
- *Les dates x, y, z, ... ne peuvent être pris en compte car nous sommes en formation (équipe incomplète)*
- *Nous convoquons les classes (ou partie de classe) suivants les types d'examen à effectuer (bilan partiel, bilan complet, ...)*
  - *En 3ème maternelle, nous effectuons un bilan complet et ne pouvons voir plus de 25 élèves sur une demi-journée à l'ensemble des classes convoquées ne peut donc rassembler plus de 25 élèves*
  - *En 1er maternelle, nous effectuons un bilan partiel et ne pouvons voir plus de 40 élèves sur une demi-journée à l'ensemble des classes convoquées ne peut donc rassembler plus de 40 élèves*
  - *...*

Les directions des établissements sont ensuite notifiées des dates de rendez-vous les concernant et sont tenus de confirmer la venue des élèves convoqués. »

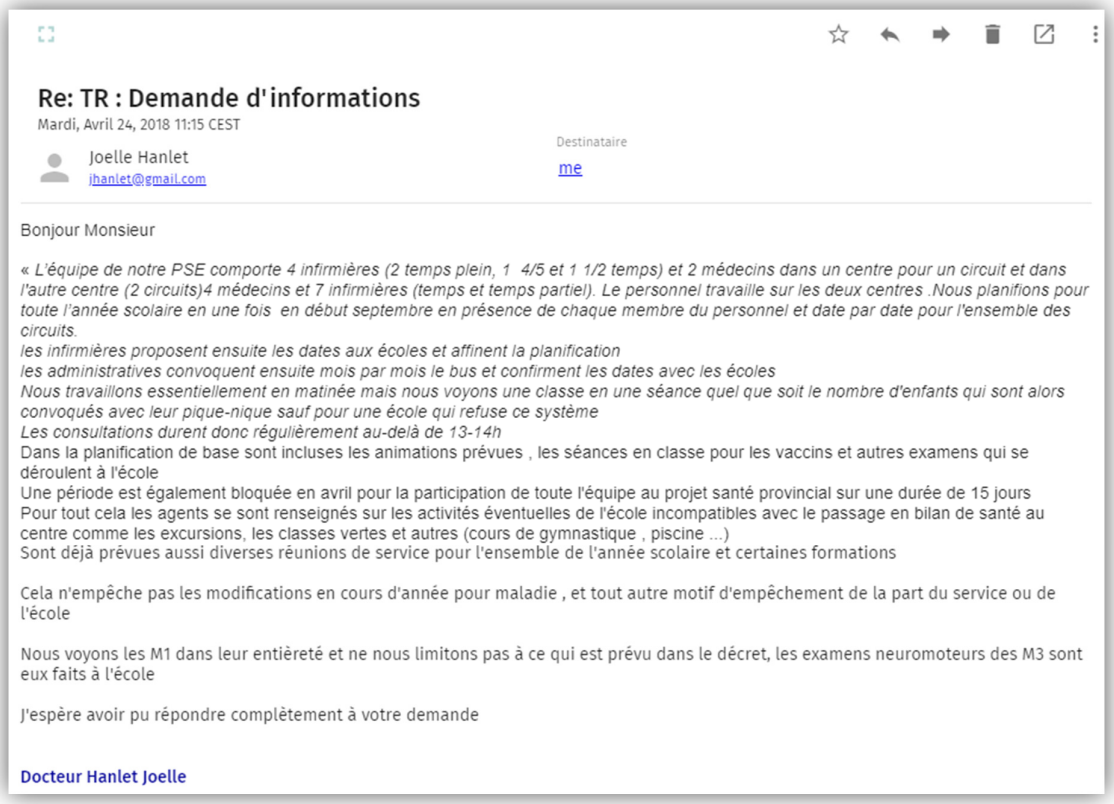
Je vous remercie de l'attention que vous voudrez bien porter à ma demande et reste à votre entière disposition pour toute question.

Cordialement.

Clément Rwanyindo.
Etudiant Master Science Informatique UNamur.

Sur les 141 demandes d’information, 10 se sont avérées être des adresses mail erronées ou expirées, deux antennes PSE ont proposé une rencontre en face à face et quatre ont adressé une réponse détaillant au mieux leur fonctionnement au quotidien.

Exemple de réponse : Le « Service provincial de promotion de la santé à l'école du Brabant wallon axe ouest » :



Re: TR : Demande d'informations

Mardi, Avril 24, 2018 11:15 CEST

Joelle Hanlet
jhanlet@gmail.com

Destinataire
me

Bonjour Monsieur

« L'équipe de notre PSE comporte 4 infirmières (2 temps plein, 1 4/5 et 1 1/2 temps) et 2 médecins dans un centre pour un circuit et dans l'autre centre (2 circuits)4 médecins et 7 infirmières (temps et temps partiel). Le personnel travaille sur les deux centres .Nous planifions pour toute l'année scolaire en une fois en début septembre en présence de chaque membre du personnel et date par date pour l'ensemble des circuits.
les infirmières proposent ensuite les dates aux écoles et affinent la planification
les administratives convoquent ensuite mois par mois le bus et confirment les dates avec les écoles
Nous travaillons essentiellement en matinée mais nous voyons une classe en une séance quel que soit le nombre d'enfants qui sont alors convoqués avec leur pique-nique sauf pour une école qui refuse ce système
Les consultations durent donc régulièrement au-delà de 13-14h
Dans la planification de base sont incluses les animations prévues , les séances en classe pour les vaccins et autres examens qui se déroulent à l'école
Une période est également bloquée en avril pour la participation de toute l'équipe au projet santé provincial sur une durée de 15 jours
Pour tout cela les agents se sont renseignés sur les activités éventuelles de l'école incompatibles avec le passage en bilan de santé au centre comme les excursions, les classes vertes et autres (cours de gymnastique , piscine ...)
Sont déjà prévues aussi diverses réunions de service pour l'ensemble de l'année scolaire et certaines formations

Cela n'empêche pas les modifications en cours d'année pour maladie , et tout autre motif d'empêchement de la part du service ou de l'école

Nous voyons les M1 dans leur entièreté et ne nous limitons pas à ce qui est prévu dans le décret, les examens neuromoteurs des M3 sont eux faits à l'école

J'espère avoir pu répondre complètement à votre demande

Docteur Hanlet Joelle

1.5.2.1. Interview

Parmi les propositions d'entrevue, c'est l'équipe du PSE de Namur ayant pour pouvoir organisateur l'ASBL SELINA qui a été rencontrée. Principalement, parce que les accompagnements scolaires de ce service s'adressent à tous les types de population, de la maternelle au supérieur.

L'entrevue avait pour objectif de :

- déterminer dans quelle mesure la réalité de terrain correspond aux désidératas des décrets
- identifier les facteurs implicites (par exemple la distance établissement/service PSE – point soulevé dans une des réponses par mail) pouvant influencer la planification des rendez-vous

Un ensemble de points (cf. *annexe 1*) avait pour se faire été préparé afin d'être sûr de couvrir le plus largement possible le contexte de fonctionnement de ce service et en dégager les éléments les plus pertinents dans le cadre de mémoire.

1.5.2.2. Bilan

Lors de l'interview, les points suivants, reflétant la réalité de terrain du PSE de Namur, ont été relevés :

- Le PSE est constitué de trois équipes (deux membres par équipe, une infirmière et une aide paramédicale) et un temps-plein administratif
- Moins de trois demi-journées par semaine sont consacrées à l'administratif, la préparation de dossier, etc.
- Chaque équipe est responsable d'un ensemble d'école

- Chaque équipe planifie ses rendez-vous avec ses écoles et dispose de son propre agenda
- Un calendrier global existe afin d'y fusionner les agendas des différentes équipes et mieux coordonner ce qui est commun. Par exemple les salles (savoir quand elle est libre pour être nettoyer, …)
- Il y a 3 salles de consultation que se partagent les 3 équipes
- La direction, ou un référent désigné par cette dernière, d'un établissement est le point de contact du service et valide les propositions de date de rendez-vous
    - Pour le fondamental (maternelle et primaire), c'est souvent le directeur
    - Pour les autres, c'est souvent une personne de l'établissement, en général un éducateur, qui se voit déléguer ces responsabilités.
- La majorité des établissements ont un jour préféré pour leur visite médicale
    - Certains établissements du supérieur demandent à ce que telle ou telle sections soient vues lors du premier semestre car les élèves partent en stage après
    - Le jour préféré est un critère arrêté qui se perpétue dans le temps
    - L'équipe PSE adapte son horaire et peut également changer le médecin responsable de l'établissement pour que ce jour soit respecté
- Il n'existe pas de lien numérique entre les établissements et le service PSE. Les listings élèves sont transmis sous format Excel et sont ensuite remontés dans le logiciel utilisé
- Les rendez-vous des différentes classes d'un établissement se planifient parfois sur toutes l'année (pas de priorité particulière) du moment que le jour préféré soit pris en compte si ce choix existe
- Les classes sont vues dans leur entièreté (pas de demi-classe par exemple) et généralement accompagnées par un éducateur
- Les rendez-vous se font par demi-journée (9h-12h, 13h-15h30)
- Les 4èmes primaires sont rencontrés dans l'établissement car ce n'est qu'un examen de vue
- Les élèves de l'enseignement spécialisé sont vus une année sur deux et non par classe
- Les élèves du supérieur sont également vus par classe, c'est-à-dire dans leur cas suivant les groupes de travail constitués par l'établissement
- Les absents sont convoqués avec d'autres classes. En fin d'année, ils viennent par groupe
- Les dossiers élèves sont autant que faire se peut préparer à l'avance afin de gagner du temps (récolte d'informations avant le jour de la visite)
- Les médecins ne sont pas des employés du service mais des indépendants auxquels le service fait appel.
- Les médecins transmettent en début d'année leur disponibilité pour l'année scolaire considérée
- Les médecins sont associés à des écoles dont l'horaire scie à leur attente (si un établissement préfère être vu le mardi c'est à un médecin ayant des disponibilités le mardi que le suivi sera confié)
- Un médecin assurera généralement d'année en année le suivi des mêmes établissements
- Des moments sont réservés aux visites d'initiation ou de suivi des vaccinations élèves :
    - déplacement dans l'école pour donner différentes doses de vaccin dans un intervalle de temps défini. Ex. : 2ème secondaire, classe 2D, visite une 1ère fois en septembre et une 2nd fois en mars
    - L'un de ces rendez-vous de vaccination peut être couplé à un bilan de santé
- Pas de moments réservés aux formations (si elles ont lieu c'est le samedi par exemple)
- Quatre journées sur l'année sont réservées à des rencontres avec d'autres services PSE du même pouvoir organisateur afin d'harmoniser la manière de travailler (modèle de document à remettre au parent, …)
- Doivent également être vus en visite médicale tous les élèves qui fréquentent pour la première fois en Belgique un établissement scolaire et qui viennent d'un pays où il y a une endémie de tuberculose
    - Listing de ces élèves transmis par l'établissement en début et mi année scolaire

    - Parfois des classes regroupant ces élèves existent (remise à niveau, …)
- La solution logicielle utilisée au sein de l'antenne PSE est propre à chaque équipe. On a donc trois bases de données, trois fois la même application, aucune information mutualisée (les données sur les locaux existant par exemple, le calendrier global de l'antenne, etc.).

A la suite de cette interview, un complément d'information a été demandé par mail afin de clarifier les données utilisées dans cette étude de cas :

- Le car ne peut accueillir plus de 30 élèves (en général c'est 20-25 élèves)
- 11 médecins collaborent avec le service PSE
- Chaque équipe effectue le suivi de 15 à 20 établissements

1.6. Positionnement final du problème

Suite à l'analyse du métier et à l'enquête de terrain menée, nous pouvons arrêter la situation suivante que nous tenterons de résoudre à l'aide de la technique de programmation déclarative.

Une équipe d'un PSE doit successivement rencontrer les étudiants des classes d'un ensemble d'établissements.

Ces rendez-vous doivent répondre aux critères suivants :

- Ils ne peuvent avoir lieu que durant les périodes scolaires (c'est-à-dire hors congé scolaire, hors weekend, hors jours fériés, hors mercredi après-midi).
- Ils doivent respecter le jour de préférences des établissements si ces derniers en ont un
- Ils ont lieu soit le matin, soit l'après-midi
- Ils ne peuvent avoir lieu un jour où le médecin responsable de l'établissement est indisponible
- Ils ne peuvent avoir lieu durant les X jours où l'équipe PSE a planifié une rencontre avec d'autres services PSE
- Ils ne peuvent avoir lieu si aucun local n'est disponible
- 1 à 3 demi-journées par semaine doivent subsister afin de permettre à l'équipe de s'occuper des aspects administratifs et de la préparation des dossiers des élèves à voir
- Une classe compte maximum 30 élèves. Plusieurs classes d'un même établissement peuvent être convoquées sans pour autant dépasser ce nombre maximum.

Dans un souci de clarté et d'efficience, nous arrêtons les points suivant :

- une classe du supérieur pourrait avoir rendez-vous le mercredi après-midi. Nous choisissons que non afin de limiter les sources de confusions inutiles.
- Toutes les plages de rendez-vous inutilisables sont renseignées en amont (congé, rendez-vous de vaccination, …)
- Bien que pour le PSE de Namur chaque équipe dispose de son local de consultation, nous supposons les locaux comme étant partagés et pouvant être en nombre inférieur au nombre d'équipe d'une antenne PSE, ici 2 locaux pour les 3 équipes. Ce choix est motivé par le fait que la quantité de locaux disponibles constitue une variable intéressante dans le cadre de cette étude et augmente la probabilité de portabilité de la solution vers d'autres services PSE.
- Nous considérons la planification sur la période scolaire 2018-2019.

1.7. Cadre conceptuel de solution automatisée

La solution automatisée de planification de rendez-vous viendra compléter le système informatique existant. Nous pouvons envisager le système suivant en vue de répondre à la problématique définie ci-dessus.

1.7.1.Les données

Les données (équipes PSE, locaux, établissements, classes et médecins) traitées par l'applicatif sont issues d'une base de données relationnelle déjà existante dans l'antenne PSE. Si cette dernière n'est pas informatisée, une base de données peut être implémentée et alimentée de ces mêmes données.

Les données relatives aux établissements sont issues des listings Excel transmis par les directions de ces dernières.

Les données relatives aux médecins sont encodées manuellement pour ce qui est des attributions d'établissements, c'est-à-dire la définition de quel médecin est responsable de quel établissement. Pour leurs agendas, leurs indisponibilités sont soit encodées manuellement soit importées dans la base de données au travers d'une opération manuelle traitant des fichiers .ics ou .ical correspondant aux exportations des médecins depuis leur calendrier individuel.

L'encodage et l'importation des données nécessitent la mise à disposition pour le service PSE d'interfaces applicatives pour effectuer ces opérations.

1.7.2.L'application

L'application de gestion des planifications de rendez-vous fournira les fonctionnalités suivantes :

- Un compte pour chaque équipe de l'antenne PSE
- Un compte pour chaque établissement. Cela afin que les directions puissent valider les propositions de rendez-vous soumises par la solution automatisée pour leur école.
- Encodage des attributions de chaque médecin
- Encodage ou importation des agendas de chaque médecin
- Si le service PSE n'est pas déjà informatisé
    - Importation des listings Excel
- Un agenda renseignant les rendez-vous planifiés

Les rendez-vous de visite médicale avec les classes des différents établissements sont générés par un automate interne à l'application.

1.7.3.L'automate

L'automate de l'application reçoit en entrée une description de la situation à traiter à l'instant t. Cette description reprend :

- les équipes existantes
- les locaux
- les établissements et leur jour de visite
- les classes de ces établissements qui n'ont pas encore de rendez-vous planifiés
- les médecins et leurs attributions ainsi qu'indisponibilités
- les dates de rendez-vous possibles de l'instant t jusqu'à la fin de l'année

L'automate renvoi les résultats (les dates de rendez-vous pour chaque classe) sous un format que l'applicatif peut traiter (.json par exemple) avant insertion de ces données dans la base de données.

L'automate est l'objet de notre intérêt dans le cadre de ce mémoire et nous nous focaliserons sur son développement. Nous suggérons cependant une solution d'implémentation des autres éléments (base de données, application web, …) une fois l'automate créé.

Chapitre 2 Les techniques de planification d'horaires

Nous analysons dans cette deuxième partie les techniques de planification pouvant répondre à la problématique soulevée.

2.1. La planification

La notion de planification peut prendre plusieurs sens suivant le contexte considéré [9, p. 3].

Le sens le plus familier consiste à associer à un ou plusieurs temps t un ou plusieurs éléments spécifiques. Ces éléments peuvent être de natures diverses. Une tâche, un évènement ou même une position (être un endroit précis au moment considéré).

Nous pouvons constater que cette tentative d'explication basée sur l'intuition utilise elle-même d'autres termes pouvant également prendre différents sens suivant le contexte envisagé. De fait, qu'est-ce qu'un évènement ? Qu'entend-on par position ?

Nous voyons cependant que si nous fixons un contexte (ou domaine), dans notre cas la planification des visites médicales des services de Promotion de la Santé à l'Ecole, une définition pourrait émerger de nos réflexions.

2.1.1. Les contextes de planification des PSE

La planification des visites médicales vise à déterminer à quelle instant t il est possible de rencontrer les élèves d'une classe afin d'effectuer un suivi de santé. Cet instant t correspond à une matinée ou après-midi d'une journée de l'année scolaire où les élèves ont effectivement cours (pas le weekend, pas les jours fériés, …).

La planification cherche à atteindre, à partir d'une situation initiale, une situation idéale répondant à la problématique que nous cherchons à résoudre.

2.1.2. La situation initiale

On a au départ la situation suivante :

- l'ensemble des classes C$(d,e)k$, $k \in [1,n]$ avec n le nombre total de classe géré par l'équipe, $e \in [1,30]$ le nombre d'élèves dans la classe et $d \in [0,5]$ le $d^{ème}$ jour de la semaine ($d = 0$ si n'importe quel jour peut convenir)
- l'ensemble des instants t, c'est-à-dire les dates possibles pour rendez-vous, D$(j,s)i$ où j est le $j^{ème}$ jour de la semaine, s est le $s^{ème}$ semestre de l'année[6] et $i \in [1,m]$ avec m le nombre maximum de date de rendez-vous possible
- L'ensemble Lc des locaux, $c \in [1,p]$ avec p le nombre maximum de locaux

2.1.3. La situation idéale

La situation idéale résultant de la planification correspond à l'ensemble de rendez-vous R(C$(d,e)ki$, D$(j,s)i$, Lc) où chaque classe C$(d,e)k$ a été associée à une date i possible de rendez-vous avec $d = j\ ssi\ d \neq 0$ tel que $\forall i, \sum e < 30$ et s est le 4ème semestre de l'année si C est une classe de 1er maternelle.

Pour atteindre cette situation idéale, nous devons **rechercher** les associations (C$(d,e)ki$, D$(j,s)i, Lc$). Cette recherche est un **plan/séquence d'actions** transitant d'un état à l'autre dans le but d'arriver à l'état final visé.

[6] Les classes de première maternelle doivent être rencontrées durant le dernier semestre de l'année scolaire

2.1.4. Définition

La **planification** de visites médicales des Services de Promotion de la Santé à l'Ecole à partir d'un ensemble de classes d'élèves, la classe ayant éventuellement un jour de visite défini, et d'un ensemble de dates données est la création d'un **ensemble de rendez-vous** où chaque classe a été **associée** à une date, dont le jour correspond à celui de la classe s'il est défini, telle que le nombre d'élèves pour chaque élément ne soit pas supérieure à 30.

2.1.5. Création/Recherche d'un planning

Disposer d'un planning consiste à rechercher des dates de rendez-vous répondant aux critères de notre problématique. « Planifier est un cas particulier de résolution de problème (recherche) »[10].

Diverses techniques de programmation permettant de résoudre des problèmes peuvent ainsi être appliquées à la planification.

2.2. La recherche opérationnelle

« La recherche opérationnelle peut se définir comme *l'ensemble des méthodes et techniques rationnelles d'analyse et de synthèse des phénomènes d'organisation utilisables pour élaborer de meilleures décisions*. » [11]

La recherche opérationnelle se prête à la résolution de problèmes :

- combinatoires
- du domaine de l'aléatoire
- concurrentiels

2.2.1. Les problèmes combinatoires

Les problèmes combinatoires se caractérisent par « le nombre astronomique d'arrangements, permutations, combinaisons » [11] possible lors de la recherche d'une réponse.

2.2.1.1. Le problème du voyageur de commerce

Afin d'illustrer la complexité des problèmes combinatoire, considérons le cas d'école suivant. Un voyageur de commerce vise à trouver, parmi un ensemble de villes dont on connait les distances entre chaque paire de ville, le chemin le plus court passant par chaque ville une et une seule fois. [12]

« Par exemple, pour 71 villes, le nombre de chemins candidats est supérieur à $5\ x\ 10^{80}$ qui est environ le nombre d'atomes dans l'univers connu. » [12]

On voit donc que les problèmes combinatoires peuvent très vite devenir difficile à résoudre voire impossible.

Généralisé, le problème du voyageur peut se formuler comme suit : Un réseau étant donné, peut-on le visiter entièrement en minimisant une certaine fonction de coût ? [13, p. 143]

Notre objectif consistant à sortir un planning de rendez-vous pour les visites médicales rentre-t-il dans cette catégorie de problème combinatoire ? Analysons à l'aide de la théorie des graphes, domaine des mathématiques jouant un rôle majeur dans la recherche opérationnel [13], notre problématique.

2.2.1.2. La coloration de graphe

En théorie des graphes, « un graphe G est un triplet (V, E, ψ) tel que :

- V est un ensemble dont les éléments sont appelés **sommets** ou **nœuds**,
- E un ensemble dont les éléments sont appelés **arêtes**

- ψ est une fonction, dite **fonction d'incidence**, qui associe à chaque arête un sommet ou une paire de sommets. » [14]

Les graphes permettent de modéliser des systèmes pouvant être représentés par des réseaux. Le réseau internet, le réseau d'amis d'un utilisateur de Facebook, le réseau de neurones d'un cerveau, etc. [14]

La coloration des graphes concerne soit les arêtes, soit les sommets et cherche à déterminer une coloration optimale de ces éléments en obtenant le minimum possible de couleurs, deux éléments adjacents ne pouvant avoir la même couleur.

Lorsqu'on considère les arêtes, le nombre minimal de couleurs recherché s'appelle l'**indice chromatique** noté « $\chi'(G)$ » [14, p. 41].

Lorsqu'on considère les sommets, le nombre minimal de couleurs recherché s'appelle le **nombre chromatique** noté « $\chi(G)$ » [14, p. 49].

« Deux problèmes algorithmiques de coloration sont les suivants :

- Problème de décision : étant donné G, un graphe, et k un entier, existe-t-il une coloration valide de G utilisant k couleurs ?
- Problème d'optimisation : étant donné G, quelle est son nombre chromatique ? » [15]

On cherche donc pour un **problème de décision** à trouver une solution qui satisfait la propriété k et pour un **problème d'optimisation** à minimiser le coup de la solution trouvé.

Dans le cadre de notre problématique de planification de visites médicales, les n rendez-vous possibles peuvent être représentés graphiquement par des intervalles sur la ligne du temps. Ces intervalles, par nature, ne se chevauchent pas. « Le graphe obtenu est dit d'*intervalles propres* » [16].

On a : $\chi(G) = \chi'(G) = k = 2$.

Notre problématique vise à associer à chaque intervalle une classe $C(d,e)ki,\ k \in [1,m], i \in [1,n]$. L'indice chromatique et le nombre chromatique n'ont pas de pertinence[7] dans la recherche d'une solution de planification pour notre problème.

Nous recherchons en fait un arrangement quelconque des classes tel que chaque classe $C(d,e)k$ soit associée à un intervalle $i \in [1,n]$ sur cette ligne du temps.

2.2.2. Les problèmes du domaine de l'aléatoire

Les problèmes impliquant l'aléatoire font référence aux problèmes où des variables initialement ignorés ou inconnues viennent mettre à mal le modèle envisagé.

Dans le cadre de la planification des visites médicales, ce peut être le fait qu'un grand nombre de rendez-vous soient aient trainés à faire l'objet d'une validation ou refus pour ensuite au milieu de l'année scolaire voir l'une au l'autre partie refuser chacun de ces rendez-vous.

Le système se verrait alors dans l'incapacité de sortir un planning étant donnée le nombre élevé de classe à traiter en vis-à-vis du peu de date restante avant la fin de l'année scolaire.

Les problèmes de l'aléatoire doivent donc être anticipés autant que faire se peut en amont de l'implémentation du système.

[7] Notre intuition nous amène à penser que les dates de rendez-vous ne sont pas liées à des notions/besoins d'adjacences. Néanmoins, intuition n'étant pas « démonstration », cela n'exclut pas la possibilité de trouver un algorithme utilisant ces concepts.

2.2.3. Les problèmes concurrentiels

Les problèmes concurrentiels combinent à la fois l'aspect combinatoire et l'aspect aléatoire [11]. De fait, les choix concurrents (au moins deux) ne peuvent être prédits.

Au niveau des planifications de rendez-vous, les locaux constituent des ressources pour lesquelles les équipes PSE ont des accès concurrentiels. Si un rendez-vous est fixé à un moment donné Di il a également lieu dans un local Lc, local qui ne pourra être usité à ce même moment Di par une autre équipe de l'antenne PSE

2.3. Les processus décisionnels de Markov

Les processus décisionnels de Markov ont pour domaine de travail la **planification dans l'incertain**. Ils utilisent les probabilités appliquées sur les actions des agents (ou acteurs) pour établir des décisions concernant le plan d'action à entreprendre.

« Les processus décisionnels de Markov (que l'on note aussi MDP) intègrent les concepts d'*état* qui résume la situation de l'agent à chaque instant, d'action (ou décision) qui influence la dynamique de l'état, de *revenu* (ou *récompense*) qui est associé à chacune des transitions d'état. Les MDP sont alors des chaînes de Markov visitant les états, contrôlées par les actions et valuées par les revenus. Résoudre un MDP, c'est contrôler l'agent pour qu'il se comporte de manière optimale, c'est-à-dire de façon à maximiser son revenu. » [17, p. 17]

Les MDP sont « définit par un quintuplet : (S, A, T, p, r) où :

- S est l'espace d'états dans lequel évolue le processus ;
- A est l'espace des actions qui contrôlent la dynamique de l'état ;
- T est l'espace des temps, ou axe temporel ;
- $p()$ sont les probabilités de transition entre états ;
- $r()$ est la fonction de récompense sur les transitions entre états. » [17, p. 18]

Les processus décisionnels de Markov constituent une réponse aux problèmes de nature stochastiques de la recherche opérationnelle.

La recherche et l'implémentation d'une solution à notre problématique à l'aide des processus décisionnels de Markov sort du cadre de ce mémoire mais pourrait constituer une piste de résolution intéressante. Surtout dès lors que notre système évoluerait vers des traitements intégrant de l'aléatoire (l'établissement a un jour préféré mais il n'est pas obligatoire de s'y tenir).

Notre système évoluerait alors vers plus d'autonomie devant prendre en compte plus de paramètres en entrée afin d'anticiper des problèmes du domaine de l'aléatoire.

Néanmoins, la création d'un tel système ne serait pas pertinente étant donné la nécessite de garder les décisions finales entre les mains du métier.

2.4. Planifier avec STRIPS

« STRIPS (ou *Stanford Research Institute Problem Solver*), est un algorithme de planification classique » [18][8] permettant de planifier dans un contexte où toutes les actions pouvant être entreprises font parties d'un ensemble fini.

« STRIPS fait partie de la catégorie des résolveurs de problème qui cherche une représentation de "modèle de monde" afin d'en trouver un dans lequel un but donné peut être atteint » [19, p. 1].

[8] Planification classique par opposition à la planification dans l'incertain

STRIPS nous demande donc de décrire le monde. Cette description s'effectue au travers de **préconditions**, d'**objets** de ce monde et d'**actions** qui peuvent être entreprises sur ces objets et des **effets** résultants.

STRIPS est basé sur la logique propositionnelle. Le monde ou domaine qu'il modélise doit pouvoir être décrit avec des objets, actions, préconditions et effets clairement identifiés.

« Pour n'importe quelle "modèle de monde", nous assumons qu'il existe un ensemble applicable d'opérateurs, chacun transformant le modèle de monde en un autre modèle de monde. La tâche du résolveur de problème est de trouver un arrangement des opérateurs qui transforme un monde initial donné en un autre qui satisfait les conditions d'un but donné » [19, p. 1□2].

Nous pouvons écrire les programmes STRIPS à l'aide du langage PDDL (Planning Domain Definition Language).

STRIPS pourrait répondre à nos attentes dans notre quête de solution pour la mise en place d'un système permettant de planifier des rendez-vous pour les services de Promotion de la Santé à l'Ecole.

Nous aurions dans ce système :

- Les préconditions suivantes :
  - Une date de rendez-vous visée est disponible
  - Une salle non réservée
- Les objets suivants :
  - Les équipes de l'antenne PSE
  - Les salles
  - Les médecins et leurs congés
  - Les établissements
  - Les classes
  - Les
- Les actions suivantes :
  - Associer une classe de première maternelle à une date de rendez-vous disponible dans le 4ème semestre en respectant le jour définit par l'établissement
  - Associer une classe de première maternelle à une date de rendez-vous disponible dans le 4ème semestre
  - Associer une classe à une date de rendez-vous disponible en respectant le jour définit par l'établissement
  - Associer une classe à une date de rendez-vous disponible
- Les effets
  - La date de rendez-vous visée non disponible
  - La salle visée non disponible
  - La classe possède une date de rendez-vous

La recherche et l'implémentation d'une solution à notre problématique à l'aide du solveur STRIPS sort du cadre de ce mémoire mais pourrait constituer une piste de résolution intéressante.

Si pas pour ce problème, pour d'autres cas de planification, notamment dans le monde de l'intelligence artificielle[9], domaine visée par cette technique de résolution de problème.

[9] Le lecteur intéressé par un exemple concret peut consulter l'exemple suivant :
http://www.primaryobjects.com/2015/11/06/artificial-intelligence-planning-with-strips-a-gentle-introduction/

2.5. La programmation déclarative

Nous appliquons pour résoudre le cas d'étude de planification une technique de programmation dite « programmation déclarative », dans l'espoir de fournir une implémentation efficiente de la planification des visites médicales d'un service PSE. Nous jugerons par la suite de la pertinence de cette technique de programmation en passant en revue ses avantages et/ou inconvénients.

2.5.1. Les paradigmes de programmation

Dans le monde de l'informatique, les paradigmes de programmations sont nombreux. Pour n'en citer que quelques-uns, on retrouve la programmation fonctionnelle, relationnelle, concurrente, etc. Les plus connus restent les programmations impérative/procédurale et orientée objet.

Tous ces paradigmes ont été inventés dans l'optique de résoudre des problèmes de traitement du monde courant, choses pour lesquelles l'informatique excelle.

La programmation déclarative est un de ces nombreux paradigmes souvent inconnus du plus grand nombre. De fait, moi-même, avant d'entamer mon master, j'en ignorais jusqu'à son existence alors qu'un des langages de programmation actuel, Prolog, basé sur ce paradigme fut créé en 1972.

2.5.2. Programmation déclarative et Intelligence artificielle

La programmation déclarative permet de répondre au manque de solution viable pour des problèmes dont l'application de techniques « classiques » de programmation (en l'occurrence impérative et/ou orienté objets) conduit à des temps de résolution hors proportions (jusqu'à des milliers d'années pour résoudre certains types de problèmes) [20, p. 103].

Un des domaines de prédilection de la programmation déclarative concerne la mise en place de systèmes experts.

« Un système expert consiste en un environnement logiciel/matériel capable

- de résoudre des problèmes requiérant une grande quantité de connaissances (de savoir-faire, d'expertise)
- d'acquérir de nouvelles connaissances en vue d'étendre ou d'adapter leur expertise
- d'expliquer leur raisonnement »[21].

Ces systèmes font donc preuve d'intelligence de par leur capacité intrinsèque évolutive une fois mise en place. En les implémentant, nous rentrons dans le monde de l'intelligence artificielle, l'intelligence pouvant se définir comme suit : « L'intelligence se base ou désigne l'ensemble des capacités mentales et cognitives d'une entité. Ces capacités lui permettent d'appréhender un environnement connu ou nouveau, s'adapter à des situations, comprendre, résoudre des problèmes, prendre une décision, apprendre. »[22, p. 12]

Dans ces systèmes experts, « tout problème est délégué au couple d'acteurs : *moteur d'inférences*, *base de règles*. Le moteur enchaîne mécaniquement les étapes de cette résolution dont la base de règles lui fournit les ressources »[20, p. 103].

Par moteur d'inférences, on entend un algorithme capable de déduire une conclusion logique à partir de la base de règles fournie.

2.5.3. Système classique de gestion et informationnel d'aide à la décision

A côté des systèmes experts, nous retrouvons des systèmes classiques de gestion et des systèmes informationnels d'aide à la décision (SIAD) que nous détaillons respectivement aux points 3.1.2 et 3.1.3.

2.5.4.Les bases de la programmation déclarative

L'idée de base de la programmation déclarative peut se formuler comme suit : « décrire le **QUOI** = ce qui est calculé et pas le **COMMENT** = comment se déroule le calcul »[23].

« Cela veut dire que, lorsqu'on implémente une solution pour un problème, au lieu de spécifier *comment* atteindre un certain but dans une situation précise, on spécifie *ce que*[10] la situation (règles et faits) et le but sont … »[24] avant d'ensuite laisser le langage de programmation déduire la solution.

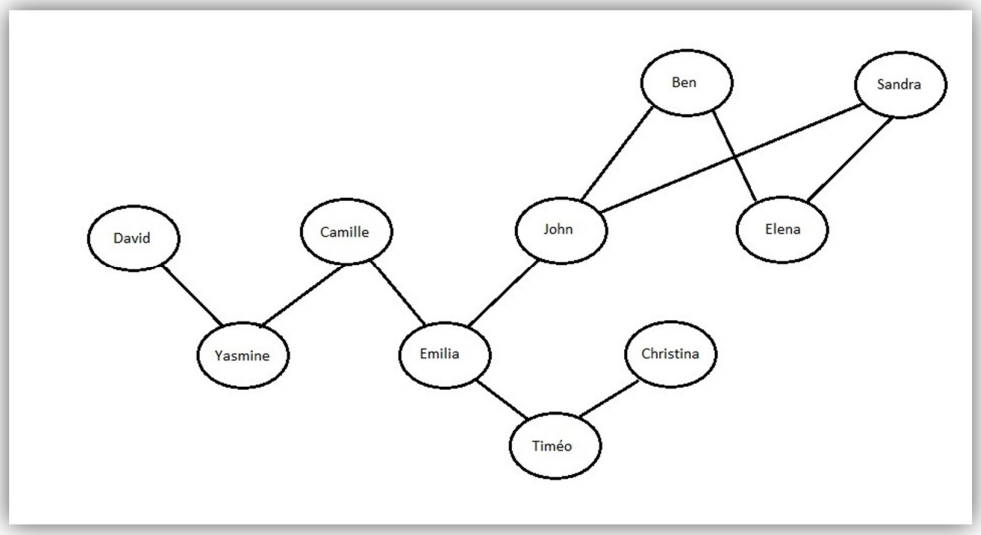
Cette idée d'une simplicité sans pareil recèle d'ingéniosité. On ne cherche plus à résoudre pas à pas une problématique en fournissant à la machine une instruction à exécuter à chaque étape. On va au contraire décrire ce qu'on veut calculer puis laisser le moteur d'inférence faire son œuvre pour nous fournir une des solutions possibles après sa déduction.

2.5.5.Prolog

Un des langages de programmation déclarative est Prolog (**Pro**grammation **log**ique). Il « est un des langages de programmation le plus largement utilisé dans la recherche en intelligence artificielle »[24].

Comme dit auparavant, en programmation déclarative, il s'agit de décrire le problème à résoudre, décrire le QUOI en énonçant des faits et règles. Une fois cette description faite, on peut alors questionner Prolog pour qu'il réponde à une question ayant trait à la description donnée.

Concrètement, nous pourrions par exemple décrire les liens de famille illustrés ci-dessous dans Prolog.



Cette illustration se traduirait par l'énonciation des faits suivants :

[10] C'est-à-dire QUOI. Traduction depuis l'anglais ou on avait initialement le terme « *WHAT* ».

family.pl

```
1   %parent(X, Y): X is parent of Y
2   parent(ben, john).
3   parent(sandra, john).
4   parent(ben, elena).
5   parent(sandra, elena).
6   parent(david, yasmine).
7   parent(camille, yasmine).
8   parent(john, emilia).
9   parent(camille, emilia).
10  parent(emilia, timeo).
11  parent(christina, yimeo).
```

Nous disposons donc d'une situation nous informant que telle ou telle personne est parent de telle ou telle autre personne.

Si nous voulons savoir quels sont les parents d'Emilia, nous pouvons questionner le système comme suit :

```
| ?- parent(X, emilia).

X = john ? ;

X = camille ? ;
```

Et si nous voulons pouvoir identifier les grands-parents de telle ou telle personne ? Nous pouvons étendre notre base de connaissance en y ajoutant des règles (une seule dans notre cas) qui vont permettre à Prolog de déduire ces relations.

```
13  %grandParent(X, Z): Y is parent of Z and X is parent of Y
14  grandParent(X,Z) :- parent(Y, Z), parent(X, Y).
```

Nous pouvons alors interroger Prolog afin d'avoir réponse à nos questions. Par exemple, demander qui est grands-parents de qui ?

```
| ?- grandParent(X, Y).

X = ben
Y = emilia ? ;

X = sandra
Y = emilia ? ;

X = john
Y = timeo ? ;

X = camille
Y = timeo ? ;
```

2.5.6.« Quel Euréka a-t-on eu ? »[11] [23, p. 11]

Nous avons utilisé un système, Prolog, capable, à partir d'une *requête* (par ex. qui est grand-parent de qui), de déterminer les conséquences logiques (Ben est grand-parent d'Emilia, …) des données (faits et règles) fournies en entrée.

Faits et règles constituent ensemble ce qu'on appelle des *clauses*. Ce terme atypique n'est pas le seul à intégrer. Prolog, de par son contexte de fonctionnement lié à la logique, définit tout un ensemble de terminologie qu'il nous faut définir ici.

2.5.7.Terminologie Prolog

Comme nous venons de le voir, un programme Prolog est un ensemble de clauses. C'est-à-dire un ensemble de faits et règles.

Les clauses sont en fait des affirmations concernant des objets du contexte considéré. Ces objets sont appelés *termes* et peuvent être de différentes natures.

Un terme peut être simple. On parle alors d'*atome*. Dans nos exemples ci-dessus, chaque personne (ben, john, sandra, …) est un atome simple.

Un terme peut être composé. On parle alors de *terme composé*, celui-ci représentant un objet structuré du contexte considéré. Ce terme composé est de la forme *foncteur(t1, …, tn)* où ti est un terme et foncteur une chaine de caractère quelconque. Par exemple, nous pourrions définir le terme composé *date(j, m, y)* pour représenter une date.

Un terme peut être une *variable*. C'est le cas de X, Y et Z dans la règle introduite dans notre exemple ci-dessus afin de définir la notion de grands-parents.

Les affirmations énoncées traduisent des relations entre termes. Ces relations peuvent être vraies ou fausses.

2.5.8.Fonctionnement de Prolog

Prolog reçoit une requête de l'utilisateur. A partir de cette requête, il va rechercher une clause dans la base de données (ensemble des faits et règles) initialement fournie qui peut se **substituer**[12] à la demande introduite et obtenir une nouvelle clause. Il réitère ensuite l'opération, jusqu'à aboutir à une conséquence logique.

La réponse fournit par Prolog est alors :

- soit « yes » si la requête initiale abouti à une conséquence logique
- soit « no » si la requête initiale n'aboutit pas à une conséquence logique
- soit l'ensemble des variables avec leurs valeurs respectives pour qu'il y ait conséquence logique

2.5.9.La programmation par contrainte

La programmation déclarative regroupe un grand nombre de sous-paradigme de programmation [25]. Nous pouvons citer pour exemple :

- La programmation logique
- Les langages dédiés (Domain specific language)

[11] Formule emprunté du cours suivi durant le cursus pour avoir suscité mon intérêt

[12] Ce mot n'est pas anodin et constitue un élément important à intégrer lors de l'apprentissage de Prolog. Nous invitons le lecteur à approfondir cette notion une fois les bases que ce mémoire essaie de transmettre acquises

- La programmation par contrainte
- ...

La programmation par contrainte se modélise au travers de la notion de « Problème de Satisfaction de Contraintes » ou CSP pour (Constraint Satisfaction Problem).

Un CSP « est un problème modélisé sous la forme d'un ensemble de contraintes posées sur des variables, chacune de ces variables prenant ses valeurs dans un domaine. De façon plus formelle, on définira un CSP par un triplet (X, D, C) tel que

- $X = \{X1, X2, \dots, Xn\}$ est l'ensemble des variables (les inconnues) du problème ;
- D est la fonction qui associe à chaque variable Xi son domaine $D(Xi)$, c'est-à-dire l'ensemble des valeurs que peut prendre Xi;
- $C = \{C1, C2, \dots, Ck\}$ est l'ensemble des contraintes. Chaque contrainte Cj est une relation entre certaines variables de X, restreignant les valeurs que peuvent prendre simultanément ces variables. » [26]

La programmation par contrainte se prête bien à la résolution de problèmes combinatoires, c'est-à-dire pour lesquelles le domaine de recherche d'une solution au problème visé est astronomique.

Appliqué à notre problématique, on a :

- $X =$ l'ensemble des classes à rencontrer
- $D(Xi) =$ l'ensemble des dates de rendez-vous possible
- $C = \{C1, C2\}$
    - $C1 = jour\ de\ la\ date\ de\ rdv = jour\ etablissement\ de\ la\ classe$
    - $C2 = si\ classe\ de\ 1ere\ maternelle, jour\ de\ rdv\ au\ 4$ème $semestre$

Chapitre 3 Modélisation de la planification à l'aide de la programmation déclarative

Nous implémentons dans cette troisième partie l'automate en charge de la planification des visites médicales pour les différentes équipes de l'antenne de Promotion de la Santé à l'Ecole.

3.1. Formalisation du système envisagé

La solution au problème de planification soulevé fonctionnera comme suit. Nous disposons au départ d'un ensemble de données qui sont :

- un ensemble de médecin et leurs indisponibilités
- l'ensemble des classes à rencontrer, chacune ayant les propriétés suivantes
    - le nombre d'élève
    - le médecin attitré
    - le jour de préférence s'il y en a un
- N locaux
- les dates d'indisponibilités du service
    - les jours de congés (weekends, fériés, vacances, …) – à inventorier
    - les dates de rendez-vous avec les autres services PSE
    - les demi-journées consacrées aux tâches administratives

Le système disposant de ces données, nous l'interrogeons et lui demandons de nous fournir un agenda de tous les rendez-vous avec les classes des établissements.

Le système répond soit par la négative, soit en fournissant une des solutions possibles c'est-à-dire la liste des dates de rendez-vous pour les visites médicales avec, y accolées, l'établissement/classe concerné.

Ces dates sont alors soumises à validation aux différentes parties (équipe PSE et établissement) et validées ou non.

Le système est automatiquement relancé tous les X temps afin de rechercher une nouvelle solution pour les dates refusées par une des parties. Le système tient compte des nouvelles données, à savoir, les dates d'indisponibilités du service correspondant au rendez-vous ayant été validées entre-temps par les 2 parties.

3.1.1. Les systèmes informatiques

Les systèmes informatiques peuvent se classifier parmi trois catégories :

- Les systèmes classiques de gestion
- Les systèmes informationnels d'aide à la décision
- Les systèmes experts (discutés au point

3.1.2. Les systèmes classique de gestion

Les systèmes classiques de gestion reçoivent en entrée des données qui sont ensuite traitées par un ensemble de processus connus et bien définit. Une fois ces traitements effectués, des résultats sont rendus par le système.

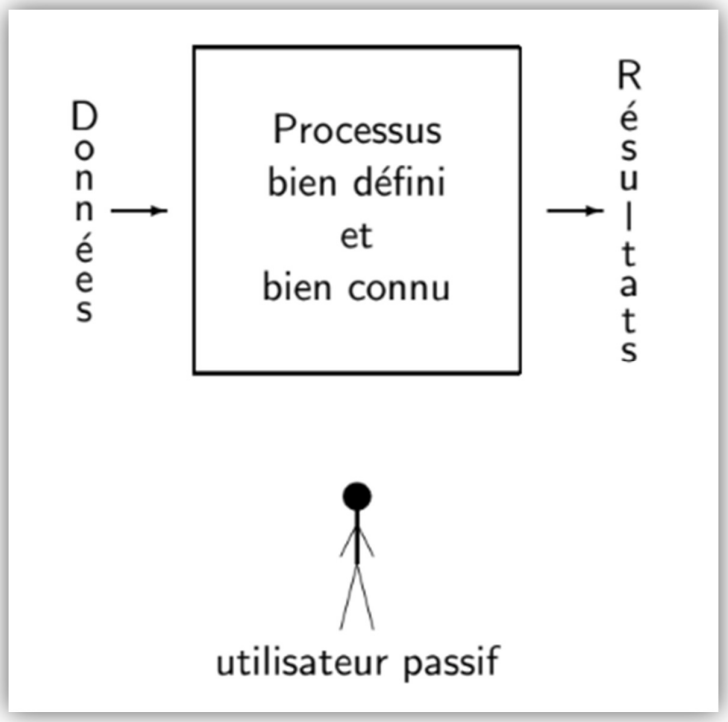


Figure 1 [21, p. 21]

L'utilisateur d'un tel système interagi avec ce dernier sans avoir une quelconque influence, d'où « utilisateur passif » sur les résultats finaux fournis par le système.

On a : $Résultats = f(Donnees)$ avec f l'ensemble des actions appliqués aux données par les processus.

Un *exemple* de système classique de gestion serait le suivant :

Imaginons un magasin offrant la possibilité à ses clients de consulter les heures d'ouverture via un site web. Le magasin voulant également tenir au courant sa fidèle clientèle met également à disposition un formulaire permettant à toute personne y accédant de s'enregistrer pour être notifier de tout changement impromptu.

On a :

- 2 processus. Un premier pour afficher la page d'information concernant les heures d'ouverture et un second permettant à l'utilisateur de s'enregistrer pour recevoir les informations de changement
- Des données
    - Pour le premier processus, aucune donnée d'entrée
    - Pour le second processus, les données demandées à l'utilisateur. Typiquement, son nom, son prénom et son adresse mail
- Des résultats
    - Pour le premier processus, l'affichage des heures d'ouverture du magasin
    - Pour le second processus, l'enregistrement des informations transmises par l'utilisateur avec message à ce dernier pour signaler que l'opération s'est bien déroulé

Appliqué au cas d'étude que nous traitons ici, le système classique de gestion serait d'application si tout le processus de planification était entièrement automatisable. Hors, nous ne pouvons pas nous passer de l'intervention des utilisateurs finaux. Ces derniers devant rester maitre de leur agenda, il faut qu'il valide d'une manière ou d'une autre la solution proposée par le système.

3.1.3.Système informationnel d'aide à la décision

Les systèmes informationnel d'aide à la décision (SIAD) reçoivent en entrée des données qui sont ensuite traitées par un ensemble de processus « non entièrement automatisable » [21, p. 22], soumises à interprétation par un utilisateur avant sortie des résultats finaux.

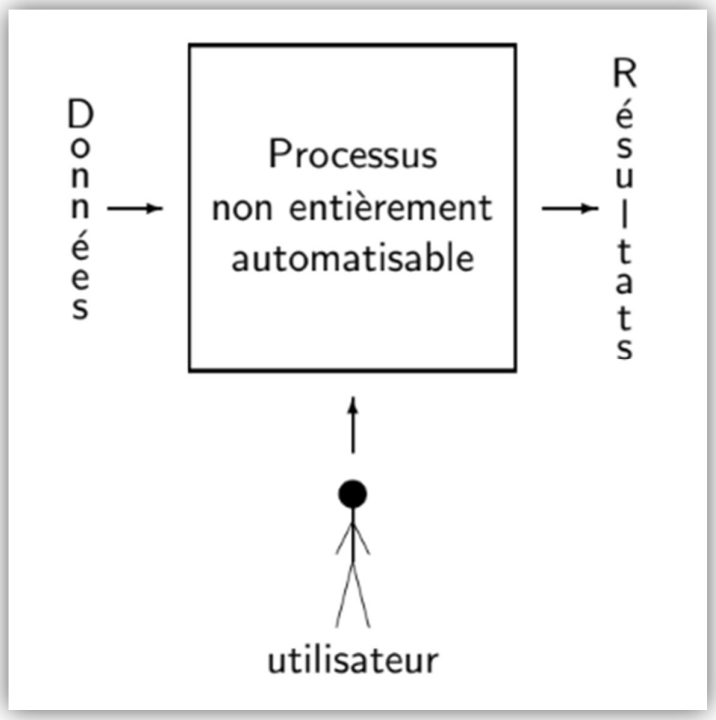


Figure 2 [21, p. 22]

Un système informationnel d'aide à la décision « est un

- système d'information
- mettant en œuvre des technologies d'information
- utilisé comme support à des activités de prise de décision relatives à des situations où il n'est ni possible, ni désirable d'automatiser l'entièreté du processus de décision »[21].

Le SIAD est donc un système qui, à l'opposé du système classique de gestion, va permettre de développer un outil dont l'entièreté des processus ne peut être automatisé et nécessite l'intervention d'un « utilisateur-décideur » [21, p. 25] qui actera pour la partie non automatisable (choix d'une option parmi un ensemble de choix possibles, validation du résultat, …) du système.

On a : $Résultats = f(Données) + d$ avec f l'ensemble des actions appliqués aux données par les processus et d l'ensemble des décisions appliquées par l'utilisateur.

Le SIAD répond parfaitement à la problématique rencontrée dans notre cas d'étude.

De fait, de nombreuses solutions sont de prime abord possible et correct. Par exemple, toutes les classes d'un même établissement, première maternelle non comprise, pourraient interchanger leur date de rendez-vous.

En considérant un petit établissement d'enseignement primaire qui aurait seulement 6 classes, on aurait comme possibilités P de solutions :

$$P = 3! = 9$$

Donc :

- $P_1 = (D_1C_2, D_2C_4, D_3C_6)$
- $P_2 = (D_1C_4, D_2C_6, D_3C_2)$
- …
- $P_9 = (D_1C_6, D_2C_4, D_3C_2)$

Où les D_i sont des dates de rendez-vous pour l'établissement d'enseignement primaire, C_2, C_4, C_6 sont les classes de l'école et $P_1, \dots, P_9$ sont toutes les permutations possibles des dates de rendez-vous des classes de l'école.

Nous calculons $3!$ et non $6!$ car les visites n'ont lieu que pour les 2ème, 4ème et 6ème primaire.

Toutes fois, dans la plus part des cas, il y a plusieurs classes pour une même année. On voit alors que le nombre de possibilités augmente très vite suivant la taille d'un établissement. Une école avec 6 classes (2 de 2ème, 2 de 4ème, …) se retrouve avec 720 possibilités de rendez-vous.

Ayant entre 15 et 20 écoles pour une équipe et les dates de rendez-vous d'un établissement à l'autre pouvant parfois s'interchanger également, on image facilement le nombre conséquent de possibilité de solution que le système peut nous fournir.

Le système va dégager une de ces solutions et la soumettre à validation auprès des utilisateurs finaux. Ces derniers ne sont pas des utilisateurs **passifs** du système mais bien **décideurs**, ce système leur sert de support à leur activité (annuelle) de prise de décisions concernant la planification de tous ces rendez-vous pour visite médicale.

3.2. Implémentation du SIAD

Nous développons ici notre programme Prolog afin de disposer d'un planning de rendez-vous sur toute l'année scolaire 2018-2019 en répondant aux exigences énoncées précédemment.

Comme présenté dans la partie « Etat de l'art », programmer avec Prolog consiste à décrire la situation rencontrée en énonçant faits et règles avant d'interroger par la suite le moteur Prolog.

3.2.1. Les équipes PSE et les locaux

Il y a 3 équipes PSE et 3 locaux. Dans un premier temps considérons qu'un local appartient à une équipe PSE comme cela est le cas pour le PSE de Namur.

```
% equipe(X): X est une équipe PSE
equipe(pse1).
equipe(pse2).
equipe(pse3).

% local(X): X est un local
local(local1).
local(local2).
local(local3).

% attribution(X,Y): l'équipe PSE X travaille dans le local Y
attribution(pse1, local1).
attribution(pse2, local2).
attribution(pse3, local3).
```

Notre principal objectif ici est de disposer d'un planning de rendez-vous avec les différentes classes des établissements suivis par chaque équipe. Nous étendrons par la suite notre programme en considérant qu'il n'y a que 2 locaux afin d'augmenter la portabilité de notre programme en anticipant ce genre de paramétrage.

3.2.2. Les établissements, leurs classes et les médecins

Chaque équipe est en charge d'un ensemble d'établissements

```
% en_charge(X,Y): X est l'équipe PSE en
% charge du suivi de la liste des établissements Y
en_charge(pse1,[etab1,etab5]).
en_charge(pse2,[etab2]).
en_charge(pse3,[etab3,etab4]).
```

Les établissements peuvent avoir un jour de visite préféré.

```
% etablissement(X,Y): X est un établissement dont
% le jour de visite est fixé à Y (j1=lundi, j2=mardi, ...)
etablissement(etab1,j3).
etablissement(etab2,j3).
etablissement(etab3,_).
etablissement(etab4,_).
etablissement(etab5,_).
```

Chaque établissement gère un ensemble de classe ayant chacune n élèves.

```
% classe(X,Y,N): X est une classe
% d'N élèves de l'établissement Y
classe(m1, etab1, 15).
classe(m2, etab1, 13).
classe(m3, etab1, 20).
classe(p1, etab1, 25).
classe(p2, etab1, 20).
classe(p3, etab1, 18).
classe(p4, etab1, 23).
classe(p5, etab1, 28).
classe(p6, etab1, 30).

classe(m1A, etab2, 14).
classe(m1B, etab2, 18).
classe(m2, etab2, 13).
classe(m3A, etab2, 19).
classe(m3B, etab2, 24).
classe(m3C, etab2, 29).
classe(p1, etab2, 30).
classe(p2, etab2, 27).
classe(p3, etab2, 24).
classe(p4, etab2, 19).
classe(p5, etab2, 28).
classe(p6, etab2, 21).
```

Les médecins collaborant avec les équipes PSE sont chacun responsable d'un ou plusieurs établissements et transmettent dès la rentrée scolaire leurs congés (données générés aléatoirement ici) au PSE.

```
% medcin(X,Y,Z): X un médecin s'occupant de la liste
% des établissements Y avec pour dates d'indisponibilités
% la liste Z (généré aléatoirement ici)
medecin(med1,[etab1,etab2],[d20181001,d20181002,d20181003,
medecin(med2,[etab3],[d20180911,d20181016,d20181130,d20181
medecin(med3,[etab4],[d20181022,d20181023,d20181024,d20181
medecin(med4,[etab5],[d20180914,d20180926,d20181002,d20181
```

3.2.3. Les dates de rendez-vous possibles

Une année comporte 365 jours[13]. L'année scolaire 2018-2019 débute le 3 sept. 2018 pour se terminer le 30 juin 2019. Dans cette plage de date, nous pouvons éliminer tous les jours fériés et tous les congés scolaires[14]. Nous pouvons également éliminer tous les weekends.

Nous obtenons la situation suivante renseignant les dates possibles de rendez-vous sur l'année scolaire.

[13] Les années bissextile comptent bien sûr 366 jours mais cela n'a pas d'incidence ici

[14] http://www.enseignement.be/index.php?page=23953

```
16  % Toutes les dates de rendez-vous possibles : une rdv correspond au
17  calendrier([rdv(d20180903,j1,_,P,L,C),rdv(d20180904,j2,_,P,L,C),rdv(
18  rdv(d20181002,j2,_,P,L,C),rdv(d20181003,j3,_,P,L,C),rdv(d20181004,j4
19  rdv(d20181105,j1,_,P,L,C),rdv(d20181106,j2,_,P,L,C),rdv(d20181107,j3
20  rdv(d20181203,j1,_,P,L,C),rdv(d20181204,j2,_,P,L,C),rdv(d20181205,j3
21  rdv(d20190107,j1,_,P,L,C),rdv(d20190108,j2,_,P,L,C),rdv(d20190109,j3
22  rdv(d20190204,j1,_,P,L,C),rdv(d20190205,j2,_,P,L,C),rdv(d20190206,j3
23  rdv(d20190311,j1,_,P,L,C),rdv(d20190312,j2,_,P,L,C),rdv(d20190313,j3
24  rdv(d20190402,j2,_,P,L,C),rdv(d20190403,j3,_,P,L,C),rdv(d20190404,j4
25  rdv(d20190502,j4,s4,P,L,C),rdv(d20190503,j5,s4,P,L,C),rdv(d20190506,
26  rdv(d20190603,j1,s4,P,L,C),rdv(d20190604,j2,s4,P,L,C),rdv(d20190605,
```

3.2.4.La génération du planning

Après avoir énoncé les faits sur lesquelles notre programme se base, nous pouvons commencer à renseigner les règles qui nous permettront de dégager un planning de rendez-vous pour chaque équipe PSE.

Nous pouvons par exemple déterminer le jour de rendez préféré de chaque classe au travers de la règle suivante :

```
109  % jour_prefere(Day, X, Y): Day est le jour préféré
110  % de la classe X de l'établissement Y
111  jour_prefere(Day,Name,Etab) :- classe(Name,Etab,_), etablissement(Etab,Day).
```

Nous reste à établir les règles nécessaires à l'établissement d'un rendez-vous pour chaque classe…

Chapitre 4 Expérimentations et Résultats

4.1. Expérimentations

Notre développement a été réalisé sur un très petit nombre de données. Nous pouvons adapter notre programme en tenant compte de la réalité du Service de Promotion de la Santé de Namur afin de tester son adéquation à la réalité de terrain aux travers de données concrètes.

4.1.1. Données de test

L'antenne PSE de Namur est composée de trois équipes. Chaque équipe dispose de son local.

Chaque équipe gère entre 15 et 20 établissements. Les établissements peuvent avoir un jour de visite (uniquement les lundis par exemple) définit pour toute l'année scolaire.

L'antenne travaille avec 11 médecins. Un médecin est responsable d'un ensemble d'école, ces écoles n'étant pas forcément suivi par la même équipe du PSE.

4.1.2. Conditions de test

A défaut de disposer de données réelles, certaines seront générés aléatoirement. En l'occurrence, nous pouvons fixer arbitrairement les données suivantes en générant des valeurs au hasard, ces valeurs respectant des bornes soit renseignées lors de l'interview avec les utilisateurs métiers soit recueilli :

- Une équipe gère X établissement, $X \in [15, 20]$
- Un médecin aura Y jours de congé[15], $Y \in [20, 33]$ et est responsable de Z écoles, $Z \in [1, 8]$

4.2. Résultats

Nous ne pouvons hélas tirer de conclusion en l'état des choses. Néanmoins, nous avons pu approfondir un sujet oh combien intéressant qui ne manquera pas de nous venir en aide notamment lorsque des problèmes « combinatoire » nous ferons face.

4.3. Intégration

La solution démontrée ci-dessus pourra s'intégrer au sein d'une application de gestion des bilans de santé des élèves des établissements gérés par un service PSE. L'architecture de cette application peut se présenter de plusieurs manières. Nous spécifions ici un choix d'implémentation basé sur les outils Docker, PostgreSQL et Python/Django. L'utilisation d'autres outils, Java/J2EE, PHP/symphony, C#/.NET Core, … est tout à fait possible également et tout aussi légitime.

4.3.1. Mise en place de l'environnement

4.3.1.1. Docker

Nous pouvons utiliser, pour la mise en place d'un environnement de test et de déploiement de notre plateforme, l'utilitaire Docker. Outil de virtualisation, ce logiciel permet de disposer assez rapidement d'un environnement similaire à une machine virtuelle appelé « container ».

Les containers sont construits à partir d'autres containers initialisés avec des caractéristiques spécifiques à une matière particulière. Ces containers de base sont appelés « image » et sont accessibles publiquement[16]. On peut ainsi créer un container constitué du cœur du système d'exploitation Ubuntu à partir de l'image « Ubuntu ». L'environnement résultant est par défaut exempt

[15] Les médecins sont indépendants. Nous nous basons sur la loi appliquée aux employés pour estimer nos bornes

[16] https://hub.docker.com/explore/

de toute fonctionnalité tel que l'interface utilisateur et va pouvoir être agrémenté de fonctionnalités suivant les besoins.

Ces containers présentent certains avantages tels que l'isolation, la portabilité, etc., mais également des inconvénients à ne pas négliger tel que l'apprentissage (nombreuses possibilités de configuration), etc.

4.3.1.2. Container Postgres de base de donnée

Une image Postgres est disponible publique et permet d'initialiser un container fournissant un moteur de base de données PostgreSQL prêt pour utilisation. Une fois initialisé, se connecter à ce container et créer la base de données et les utilisateurs nécessaires.

```
C:\Users\clement\Documents\Devel
λ docker exec -it atiapppg bash
root@7f85394f24a9:~# psql -h localhost -U postgres
psql (10.2 (Debian 10.2-1.pgdg90+1))
Type "help" for help.

postgres=# create database mydb;
CREATE DATABASE
postgres=# CREATE USER mydbuser WITH LOGIN NOSUPERUSER CREATEDB NOCREATEROLE NOINHERIT NOREPLICATION CONNECTION LIMIT -1 PASSWORD 'mydbpwd';
CREATE ROLE
postgres=# grant all privileges on database mydb to mydbuser;
GRANT
postgres=#
```

4.3.1.3. Container applicatif

Une image Python est disponible publiquement et permet d'initialiser un container avec ce langage de programmation installé et fonctionnel. Lors de l'installation, veiller à paramétrer ce dernier pour qu'il se connecte au container de base de données initialisé précédemment.

```
C:\Users\clement.rwanyindo\Documents\Devel
λ docker run -v c:\users\clement\document\devel\:/data/ -w /data -p 80:8000 -d -it --link mydb --name myapp python
```

Une fois initialisé, se connecter à ce container et installer le framework web Django à l'aide de l'utilitaire PIP fourni de base. Notre environnement est finalisé et nous pouvons commencer l'implémentation.

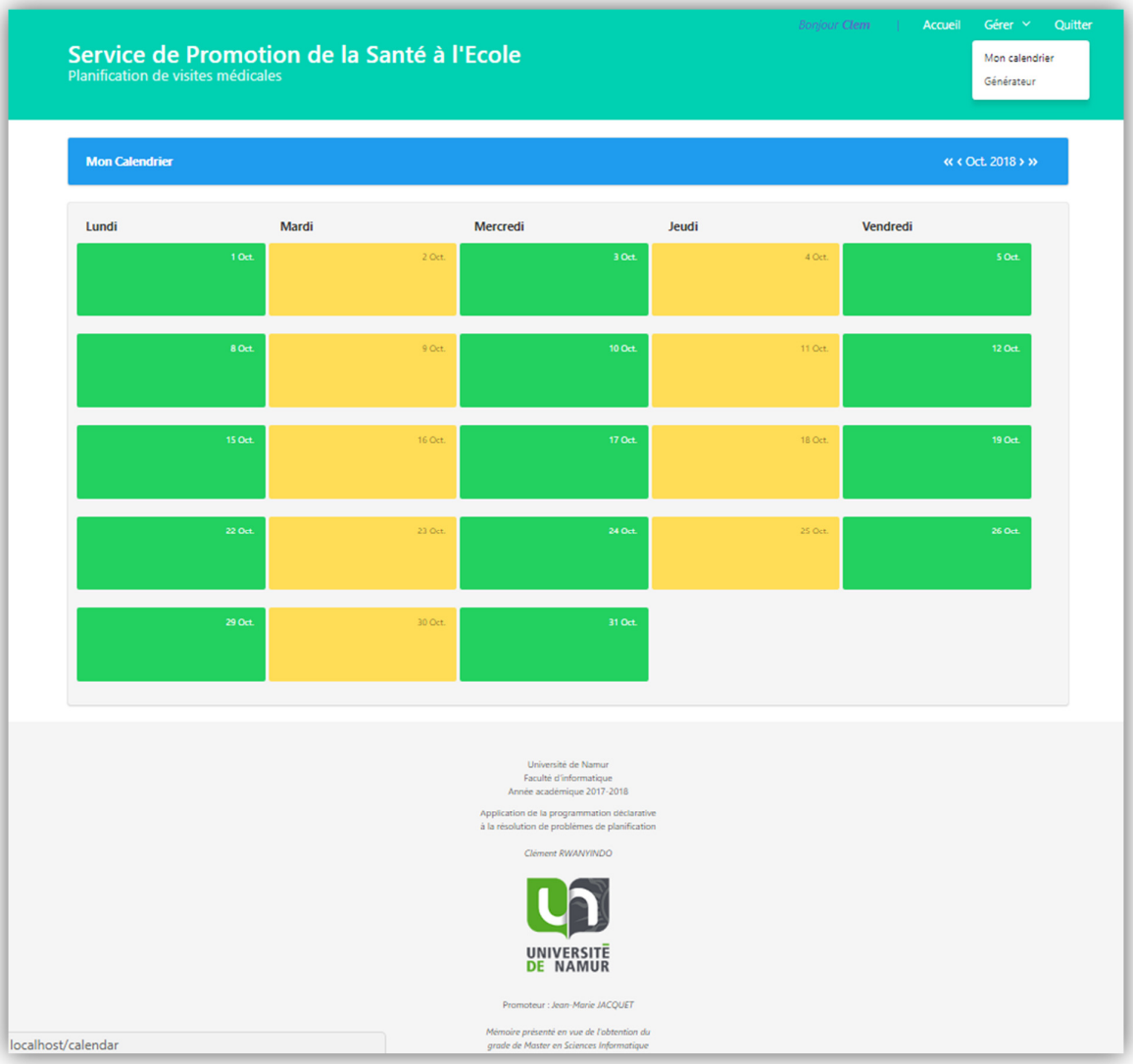
4.3.2. Développement de l'application

4.3.2.1. Conception de la base de donnée

Nous pouvons concevoir la base de données en prenant en compte les informations énoncées lors du positionnement du problème (cf. point 2.5) à l'aide de l'outil de modélisation DB-main.

4.3.2.2. Implémentation

Au niveau de l'interface, il nous faudra disposer d'un calendrier nous offrant la possibilité de naviguer dans le temps afin de visualiser les rendez-vous planifiés.



Nous pouvons également mettre en place une page « Générateur » offrant entre autre la fonctionnalité d'activation de la planification.

4.3.2.3. Intégration du générateur

Nous permettons ici à l'utilisateur de lancer la procédure de planification des rendez-vous. L'exécution de cette fonctionnalité faisant appel à la solution développée auparavant se déroulera comme suit :

- Accès à la base de données pour y rechercher toutes les données utiles (établissement, jour de préférence, classe, locaux de visite, …)
- Création de la base de règles à partir de ces données
- Activation du moteur d'inférences
- Transmission de cette base au moteur d'inférences
- Lancement du moteur d'inférences
    - Sortie des résultats
- Récupération des résultats
    - Si résultat
        - Injection des résultats dans la base de données

        - Redirection vers le calendrier
    - Sinon afficher le message « Pas de solution trouvée »

Si des rendez-vous ont été planifiés, le référent de chaque établissement est notifié afin de valider ou non les dates de rendez-vous proposées pour son établissement.

Une fois activée, cette procédure sera relancée automatiquement tous les X temps, tant que toutes les classes des établissements n'auront pas de rendez-vous fixés. Cette relance automatique ne prendra en compte que les rendez-vous faisant l'objet d'un refus du rendez-vous proposé précédemment ainsi que des nouvelles dates d'indisponibilités dues aux rendez-vous acceptés entretemps.

L'intégration de ce générateur peut se faire en appelant, depuis le langage Python, l'exécutable Prolog auquel les données d'entrées éventuellement recalculées sont fournies ou en utilisant des librairies telles que PyLog, pyswip, etc.

4.4. Aller plus loin

Nous pourrions pousser cette étude plus loin en analysant les intéressants points suivant

- Complexité algorithmique / Temps d'exécution
    - Quelle est le temps d'exécution du programme écrit en prolog ?
    - Est-ce un meilleur temps que celui que nous aurions eu au travers des paradigmes de programmation impératives et/ou objets ?
- Evolutivité vers d'autres techniques l'intelligence artificielle déduction de nouvelles règles
    - La programmation déclarative est une technique d'intelligence artificielle parmi tant d'autres
    - Dans quelle mesure pourrions-nous rendre notre système plus intelligent ? Par exemple, au fur et à mesure des années, des patterns vont, et nous pourrions en tirer certaines utilités
- Meilleur méthode d'intégration : appel à un exécutable Prolog externe ? librairie PyLog ?
    - La méthode d'intégration envisagée au point 4.4 est-elle suffisante
    - Dans quelle mesure est-ce plus intéressant d'envisager l'utilisation de framework pour intégrer la programmation déclarative au langage de base utilisé ?
- Portée
    - La problématique que nous avons traitée ici peut se transposer à d'autres contextes de planification :
        - planification dans les hôpitaux, planification dans les lieux de travail, etc.
        - Dans quelles mesures peut-on généraliser notre étude

Chapitre 5 Conclusions

Nous tirons conclusion des analyses et développements menés jusqu'ici.

5.1. Résultats de notre étude

Nous avons, au travers du cas d'étude envisagé, mis en application une technique de programmation.

Bien qu'une solution concrète au problème envisagée n'ait pu émerger, nous avons découvert le positionnement de cette technique dans le domaine de la planification par rapport à d'autres techniques.

Nous avons également pu voir son utilité face à des problèmes de nature combinatoire.

Les nombreuses recherches nous ont également fait découvrir ces autres techniques qu'il sera opportun d'utiliser lorsque le contexte s'y prêtera.

5.2. Personnellement

Si la démarche d'analyse et d'étude adoptée tout au long de ce mémoire entendait initialement répondre aux attentes du corps académique, c'est avec beaucoup de surprise et joie que ces méthodologies ont été découvertes.

Cela m'a réellement permis permis de développer un esprit plus critique et plus précis lors du suivi d'une sujet.

Cette démarche, bien que nouvelle et difficile d'apprentissage, fut très fructifiante et j'espère progresser encore plus dans ce domaine.

Chapitre 6 Bibliographie

[1] Banque Carrefour de la législation, *Loi sur l'inspection médicale scolaire*. 1964.

[2] Conseil de la Communauté Française, *Décret portant organisation de la promotion de la santé en Communauté française*. 1997.

[3] Alliance nationale des mutualités chrétiennes, « Réforme de l'inspection médicale scolaire », nº 165, p. 3.

[4] Conseil de la Communauté Française, *Décret relatif à la promotion de la santé à l'école*. 2001.

[5] Fédération Wallonie-Bruxelles, « L'organisation générale de l'enseignement », *Le portail de l'enseignement en Fédération Wallonie-Bruxelles*. [En ligne]. Disponible sur: http://www.enseignement.be/index.php?page=25568&navi=2667. [Consulté le: 22-mars-2018].

[6] ONE - Office de la Naissance et de l'enfance, « 6ème réforme de l'Etat », *ONE - Office de la Naissance et de l'enfance*, 18-nov-2014. [En ligne]. Disponible sur: http://www.one.be/actualites-one/details-actualites-one/6eme-reforme-de-l-etat/. [Consulté le: 11-avr-2018].

[7] Parlement de la Communauté française, *Décret relatif aux missions, programmes et rapport d'activités des Centres psycho-médico-sociaux*. 2016.

[8] Gouvernement de la Communauté française, *Arrêté du Gouvernement de la Communauté française fixant les fréquences, le contenu et les modalités des bilans de santé, en application du décret du 20 décembre 2001 relatif à la promotion de la santé à l'école*. 2002.

[9] S. M. LaValle, *Planning algorithms*. Cambridge University Press, 2006.

[10] Stuart Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach*, 3rd éd. Pearson, 2010.

[11] Robert Faure, Bernard Lemaire et Christophe Picouleau, *Précis de recherche opérationnelle. Méthodes et exercices d'application*, 7e éd. Paris: DUNOD, 2014.

[12] « Problème du voyageur de commerce », *Wikipédia*. 22-mai-2018.

[13] F. Meunier, « Introduction à la recherche opérationnelle ». educnet, 14-sept-2016.

[14] R. Lambiotte, « Théorie des graphes ». 2016.

[15] « Coloration de graphe », *Wikipédia*. 21-avr-2018.

[16] F. Gardi, « Planification d'horaries de travail et colorations de graphes », présenté à le 5ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision, Avignon, France, 2003.

[17] Olivier Buffet et Olivier Sigaud, *Processus décisionnels de Markov en Intelligence Artificielle - Volume 2, Méthodes avancées et applications*, vol. 2, 2 vol. Hermes Science Publications, 2008.

[18] « STRIPS », *Wikipédia*. 13-août-2016.

[19] R. E. Fikes et N. J. NHsson, « STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving' », vol. 2, issues 3-4, p. 189-208, 1971.

[20] G. Caplat, *Modélisation cognitive et résolution de problèmes*, Presses polytechniques et universitaires romandes. CH - 1015 Lausanne, 2002.

[21] J.-M. Jacquet, « Technique d'Intelligence Argificielle ». 2017.

[22] J.-M. RODRIGUEZ, *Intelligence Artificielle et Cognitive Business*, Eni. 2018.

[23] J.-M. Jacquet, « Techniques d'Intelligence Artificielle : programmation déclarative ». 2015.

[24] U. Endriss, « An Introduction to Prolog Programming ». 01-nov-2016.

[25] « Declarative programming », *Wikipedia*. 30-mai-2018.

[26] « Cours de programmation par contraintes ». [En ligne]. Disponible sur: https://perso.liris.cnrs.fr/christine.solnon/Site-PPC/e-miage-ppc-som.htm. [Consulté le: 31-mai-2018].

[27] ONE - Office de la Naissance et de l'enfance, *Organigramme_ONE_2016*. .

Chapitre 7 Annexes

Nous retrouvons les annexes suivantes:

1. Les questions préparées pour l'entrevue avec le PSE de Namur
2. Le code Prolog fournissant les rendez-vous de visites médicales

Annexe 1 : Interview du PSE Namur

- Le service PSE
  - Le personnel
    - Equivalent TP
    - Nombre infirmières et médecins
    - Nombre d'administratifs
  - Les équipes
    - Combien ?
    - Rôles ?
    - Echange de personnelles ?
  - Les établissements
    - Interlocuteur : Direction pour la planification ? Enseignant pour les modalités pratiques ?
    - Type de population d'élève : maternelle, primaire, fondamentale, …
    - Transmission des listings élèves → impact sur la possibilité de planification
  - Organisation interne
    - PO : centre de décision, finances, …
    - Calendrier personnel, d'équipe, du PSE ?
    - Programmes utilisés :
      - Outil tel que DOSMED, OmniPro, IMS+, …
      - Outlook, Google calendar, …
    - Liberté d'ajustement de ce qui est planifié
- La planification
  - Obligations légales
    - 70% du travail du service
    - Voir tous les élèves tels que prévu par la loi
      - → Par ordre de priorité : P6, P4, P2, P1r, M3, M2r, M1 (3ème trimestre)
      - → Secondaire, spécialisé, supérieure, …
    - Quid si impossible de les voir tous ?
      - Augmentation du personnel
      - Statistique ?
      - Intérêt d'un outil de planification ?
      - …
  - Quand
    - fin d'année scolaire pour l'année suivante
    - au fur et à mesure du temps
      - combien de temps à l'avance
  - Comment
    - Convocation envoyé à la direction et/ou à l'enseignant ?
    - Quantité de retour négatif ?
  - Créneau horaire
    - Demi-journée : 9h-12h et 13h-16h ?
    - Autres ?
- Les contraintes
  - Le type d'examen (bilan partiel/total) fct de la classe convoquée
  - Nombre d'élèves qu'on peut recevoir
  - Distance école/PSE
  - Vaccins à faire ? → le sait-on avant
  - Primo-arrivant ?
- Autres facteurs ?

Annexe 2 : Structure de l'ONE

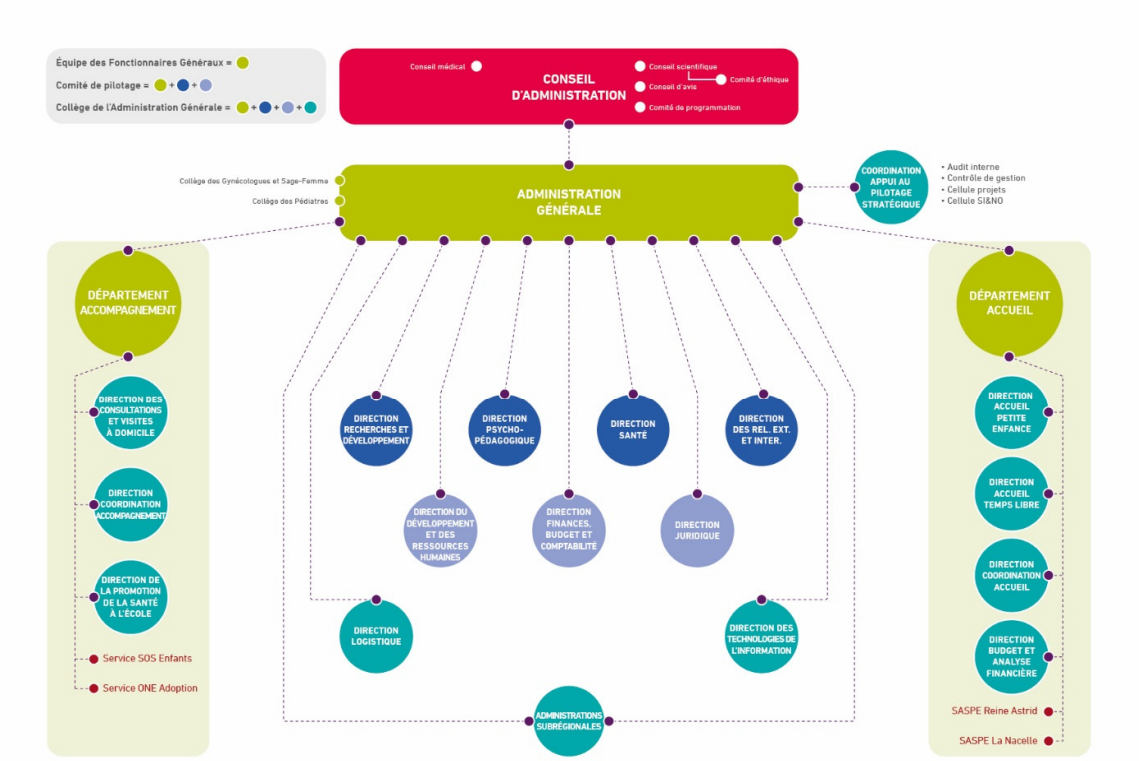
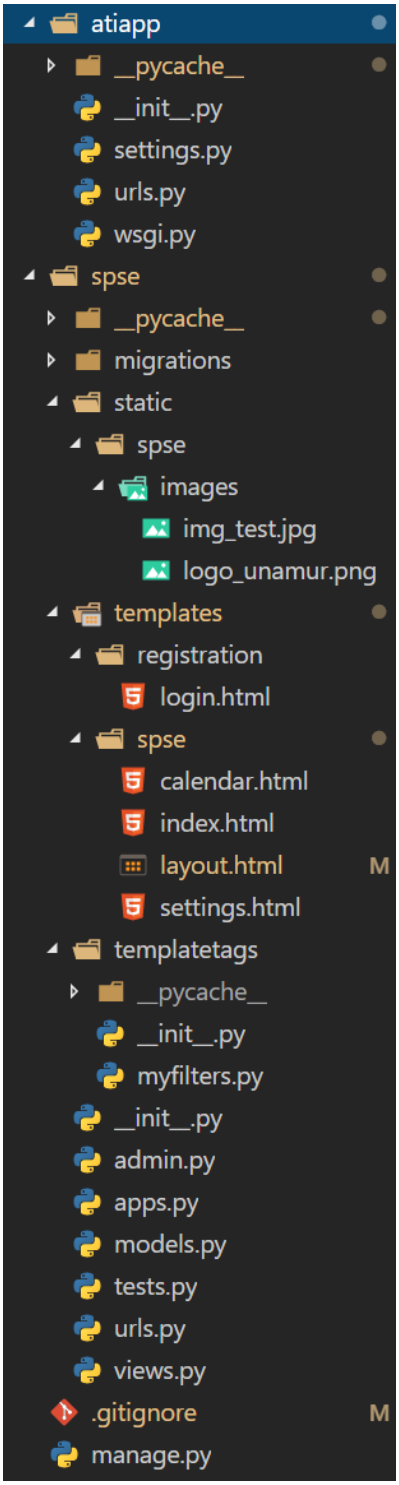


Figure 3 : Structure de l'ONE [27]

Annexe 3 : Code squelette de l'application

Structure



Settings.py

```
# Static files (CSS, JavaScript, Images)
```

```
# https://docs.djangoproject.com/en/2.0/howto/static-files/

STATIC_URL = '/static/'

STATIC_ROOT = os.path.join(BASE_DIR, "static")

LOGIN_REDIRECT_URL = '/'
LOGOUT_REDIRECT_URL = '/'
```

Urls.py

```
from django.contrib import admin
from django.urls import path, include
from django.conf.urls.static import static
from django.conf import settings

urlpatterns = [
    path('', include('spse.urls')),
    path('admin/', admin.site.urls),
    path('accounts/', include('django.contrib.auth.urls'))
] + static(settings.STATIC_URL, document_root=settings.STATIC_ROOT)
```

Spse > Templates > registration > login.html

```
{% extends 'spse/layout.html' %}

{% block title %} - Login {% endblock %}


{% block login %}

    {% if next %}
        <article class="message is-danger">
            <div class="message-header">
                <p>Erreur!</p>
                <button class="delete" aria-label="delete"></button>
            </div>
            <div class="message-body">
                Vous ne disposez pas des droits d'accès à cette page!
            </div>
        </article>
    {% endif %}

    <div class="block">
        <div class="columns is-centered">
            <div class="column is-one-third">
                {% if form.errors %}
```

```
                    <article class="message is-warning">
                        <div class="message-header">
                            <p>Erreur!</p>
                            <button class="delete" aria-
label="delete"></button>
                        </div>
                        <div class="message-body">
                            L'identifiant ou mot de passe renseignés est
incorrect!
                        </div>
                    </article>
                {% endif %}
                <div class="card">
                    <header class="card-header">
                        <p class="card-header-title is-info">Application
SPSE</p>
                        <a href="#" class="card-header-icon" aria-label="more
options">
                        <span class="icon">
                            <i class="fas fa-angle-right" aria-
hidden="true"></i>
                        </span>
                        </a>
                    </header>
                    <form method="post" action="{% url 'login' %}">
                        {% csrf_token %}
                        <div class="card-content">
                            <div class="content">
                                <div class="field">
                                    <label class="label">Identifiant</label>
                                    <div class="control has-icons-left has-
icons-right">
                                        <!--input class="input" type="text"
placeholder="Entrez votre identifiant"-->
                                        {{ form.username }}
                                        <!--span class="icon is-small is-
left">
                                            <i class="fas fa-user"></i>
                                        </span>
                                        <span class="icon is-small is-right">
                                            <i class="fas fa-check"></i>
                                        </span-->
                                    </div>
                                    <!--p class="help is-success">Identifiant
...</p-->
                                </div>
                                <div class="field">
                                    <label class="label">Mot de passe</label>
```

```
                                    <div class="control has-icons-left has-
icons-right">
                                        <!--input class="input"
type="password" placeholder="Entrez votre mot de passe"-->
                                        {{ form.password }}
                                        <!--span class="icon is-small is-
left">
                                            <i class="fas fa-lock"></i>
                                        </span>
                                        <span class="icon is-small is-right">
                                            <i class="fas fa-check"></i>
                                        </span-->
                                    </div>
                                    <!--p class="help is-success">Mot de passe
...</p-->
                                </div>
                                <div class="field">
                                    <label class="label content is-
small"> </label>
                                    <div class="control">
                                        <button type="submit" class="button
is-primary">Se connecter</button>
                                        <input type="hidden" name="next"
value="{{ next }}"/>
                                    </div>
                                </div>
                            </div>
                            <!--footer class="card-footer">
                                <a href="#" class="card-footer-item">Se
connecter</a>
                            </footer-->
                        </form>
                    </div>
                </div>
            </div>
        </div>
{% endblock %}
```

Spse > Template > spse > calendar.html

```
{% extends 'spse/layout.html' %}

{% block title %} {{ title }} {% endblock %}

{% block page %}
    <!--section class="section"-->
        <div class="container">
```

```
            <!--div class="block">
                {{ days }}
            </div-->
            <div class="tile is-ancestor is-vertical">
                <div class="tile is-parent">
                    <div class="tile is-child box notification is-info">
                        {% load myfilters %}
                        <p class="content">
                            <strong>Mon Calendrier</strong>
                            <span style="float:right">
                                <a href="/calendar/{{year|add:-1}}/{{month}}/"
style="text-decoration:none" title="Reculer d'un an">
                                    <i class="fa fa-angle-double-left"></i>
                                </a>
                                {% if month == 1 %}
                                    <a href="/calendar/{{year|add:-1}}/12/"
style="text-decoration:none">
                                {% else %}
                                    <a href="/calendar/{{year}}/{{month|add:-
1}}/" style="text-decoration:none">
                                {% endif %}
                                    <i class="fa fa-angle-left"></i>
                                </a>

                                {{ month|month_name }}. {{ year }}

                                {% if month == 12 %}
                                    <a href="/calendar/{{year|add:1}}/1/"
style="text-decoration:none">
                                {% else %}
                                    <a
href="/calendar/{{year}}/{{month|add:1}}/" style="text-decoration:none">
                                {% endif %}
                                    <i class="fa fa-angle-right"></i>
                                </a>
                                <a href="/calendar/{{year|add:1}}/{{month}}/"
style="text-decoration:none" title="Avancer d'un an">
                                    <i class="fa fa-angle-double-right"></i>
                                </a>
                            </span>
                        </p>
                    </div>
                </div>
                <div class="tile is-parent centered">
                    <br/>
                    <div class="tile is-child box notification is-light">
                        <!--
                            Titre des colonnes pour une semaine de travail
                            Mettre 7 si on veut afficher du lundi au dimanche
```

```
                        -->
                        {% with ''|center:5 as daysTitle %}
                            <div class="columns">
                                {% for _ in daysTitle %}
                                    <div class="column">
                                        <strong>{{ forloop.counter|day_name
}}</strong>
                                    </div>
                                {% endfor %}
                            </div>
                        {% endwith %}
                        {% for weeks in days %}
                            <div class="columns">
                                {% for day in weeks %}
                                    <!--
                                        Affichage des jours de la semaine de
travail
                                        Si on veut afficher la semaine
complète, enlever le if et rajouter 'is-dark' 2x au filtre cycle !
                                    -->
                                    {% if not forloop.counter|divisibleby:6
and not forloop.counter|divisibleby:7 %}
                                        <div class="column {% if day != 0 %}
notification {% endif %} {% cycle 'is-success' 'is-warning' 'is-success' 'is-
warning' 'is-success' %}">
                                            {% if day != 0 %}
                                                <h1 class="subtitle">
                                                    <span class="content is-
small" style="float:right">
                                                        {{ day }} {{
month|month_name }}.

<br/><br/><br/><br/>
                                                    </span>
                                                </h1>
                                            {% endif %}
                                        </div>
                                    {% endif %}
                                     
                                {% endfor %}
                                <div></div>
                            </div>
                        {% endfor %}
                    </div>
                    <br/>
                </div>
            </div>
        </div>
        <br/>
```

```
    <!--/section-->
{% endblock %}
```

Spse > Template > spse > index.html

```
{% extends 'spse/layout.html' %}

{% block title %} {{ title }} {% endblock %}

{% block page %}
    <!--section class="section"-->
        <div class="container">
            <div class="tile is-ancestor is-vertical">
                <div class="tile is-parent">
                    <div class="tile is-child box notification is-info">
                        <p class="content"><strong>Tableau de
bord</strong></p>
                    </div>
                </div>
                <div class="tile">
                    <div class="tile is-4 is-vertical is-parent">
                        <div class="tile is-child box notification is-
warning">
                            <p class="subtitle">Urgences</p>
                            <p class="content"></p>
                        </div>
                        <div class="tile is-child box notification is-
success">
                            <p class="subtitle">Prochainement</p>
                            <p class="content"></p>
                        </div>
                    </div>
                    <div class="tile is-8 is-parent">
                        <div class="tile is-child box notification is-light"
style="background-color:#dbd5db">
                            <p class="subtitle">Mes rendez-vous</p>
                            <p class="content" style="text-align:justify">

<br/><br/><br/><br/><br/><br/><br/><br/><br/>
                            </p>
                        </div>
                    </div>
                </div>
            </div>
        </div>
        <br/>
    <!--/section-->
{% endblock %}
```

Spse > Template > spse > layout.html

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>SPSE {% block title %}{% endblock %}</title>
    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/bulma/0.6.2/css/bulma.min.css">
</head>
<body>

    <!-- Hero section -->
    <section class="hero is-primary">
        <div class="hero-body">
            <div class="container">
                <h1 class="title">
                    Service de Promotion de la Santé à l'Ecole
                </h1>
                <h2 class="subtitle">
                    Planification de visites médicales
                </h2>
            </div>
        </div>
    </section>

    <div class="block">
        {% if user.is_authenticated %}
                <nav class="navbar is-transparent is-fixed-top is-primary">
                    <div id="navbarExampleTransparentExample" class="navbar-
menu">
                        <div class="navbar-start"></div>
                        <div class="navbar-end" style="margin-right:35px">
                            <span class="navbar-item" style="color:rgb(93,
112, 196)">
                                <i>Bonjour <b>{{ user.first_name }}</b></i>
        |
                            </span>
                            <a class="navbar-item" href="/">
                                Accueil
                            </a>
                            <div class="navbar-item has-dropdown is-
hoverable">
                                <a class="navbar-link" href="#">
```

```
                                Gérer
                            </a>
                            <div class="navbar-dropdown is-boxed">
                                <a class="navbar-item" href="/calendar">
                                    Mon calendrier
                                </a>
                                <a class="navbar-item" href="/settings">
                                    Paramètres
                                </a>
                            </div>
                        </div>
                        <a class="navbar-item" href="/accounts/logout/">
                            Quitter
                        </a>
                    </div>

                </div>
            </nav>
        {% endif %}
    </div>

    {% if user.is_authenticated %}
        <!--
            Bloque accessible uniquement aux utilisateurs identifiés.
        -->
        {% block page %} {% endblock %}
    {% else %}
        <!-- Si l'utilisateur n'est pas authentifié, il est redirigé vers
/accounts/login/ ou ce bloque est complété -->
        {%block login %} {% endblock %}
    {% endif %}

    <!-- Footer section -->
    <footer class="footer">
        <div class="container">
            <div class="content has-text-centered">
            <p class="content is-small">
                Université de Namur<br/>
                Faculté d'informatique<br/>
                Année académique 2017-2018<br/>
            </p>
            <p class="content is-small">
                Application de la programmation déclarative<br/>à la
résolution de problèmes de planification<br/><br/>
                <i>Clément RWANYINDO</i><br/>
            </p>
            <p class="content is-small">
                {% load static %}
```

```
                <img src="{% static 'spse/images/logo_unamur.png' %}"
alt="Logo1"/><br/><br/>
                <!--img src="static/spse/images/logo_unamur.png" alt="Logo2"/-
->
            <!--/p>
            <p class="content is-small"-->
                Promoteur : <i>Jean-Marie JACQUET</i><br/><br/>
                <i>Mémoire présenté en vue de l'obtention du<br/>grade de
Master en Sciences Informatique</i>
            </p>
            <!--p class="content is-small">
                Themed throught Bulma by <a href="https://jgthms.com">Jeremy
Thomas</a>. Source code of Bulma licensed
                <a href="http://opensource.org/licenses/mit-
license.php">MIT</a>. The website content
                is licensed <a href="http://creativecommons.org/licenses/by-
nc-sa/4.0/">CC BY NC SA 4.0</a>.
            </p-->
            </div>
        </div>
    </footer>

    <script defer
src="https://use.fontawesome.com/releases/v5.0.0/js/all.js"></script>
</body>
</html>
```

Spse > Template > spse > settings.html

```
{% extends 'spse/layout.html' %}

{% block title %} {% endblock %}

{% block page %}
    <div class="container">
        <div class="tile is-ancestor is-vertical">
            <div class="tile is-parent">
                <div class="tile is-child box notification is-info">
                    <p class="content"><strong>Paramètres</strong></p>
                </div>
            </div>
            <div class="tile is-parent centered">
                <div class="tile is-child box notification is-light">
                    <p class="content"><br/><br/><br/><br/><br/><br/></p>
                </div>
            </div>
            <br/>
        </div>
```

```
    </div>
{% endblock %}
```

Spse > templatetags > myfilter.py

```
from django import template
import calendar as calend

register = template.Library()

@register.filter
def month_name(month_number):
    return calend.month_abbr[month_number]

@register.filter
def day_name(day_number):
    days = {1 : 'Lundi', 2: 'Mardi', 3: 'Mercredi', 4: 'Jeudi', 5: 'Vendredi',
6 : 'Samedi', 7 : 'Dimanche'}
    return days[day_number]
```

Spse > apps.py

```
from django.apps import AppConfig


class SpseConfig(AppConfig):
    name = 'spse'
```

spse > urls.py

```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.index, name='index'),
    path('calendar/', views.calendar, name='calendar'),
    path('calendar/<int:year>/<int:month>/', views.calendar, name='calendar'),
    path('settings/', views.settings, name='settings')
]
```

Spse > views.py

```
import calendar as cal
from django.shortcuts import render, redirect
from django.utils import timezone

# Create your views here.
```

```
def index(request):
    if not request.user.is_authenticated:
        return redirect('/accounts/login/')
    context = {
        'title' : '- accueil'
    }
    return render(request, 'spse/index.html', context)


def calendar(request, year=timezone.now().year, month=timezone.now().month):
    y = year
    m = month
    days = cal.monthcalendar(y, m)
    context = {'title' : '- Calendrier', 'days' : days, 'month' : m, 'year' :
y}
    return render(request, 'spse/calendar.html', context)

def settings(request):
    return render(request, 'spse/settings.html')
```